

Kritische Infrastrukturen: Systeme von wesentl. Bed. zur Aufrechterhaltung wichtiger gesellsch. Funktionen, Gesundheit, Sicherheit, wirtsch. & sozialen Wohlenergehens der Bevölkerung
 ↳ z.B. Energie, Wasser, Internet, Verkehr

Aufbau des Internets

Komponentensicht: Internet = Rechnernetz, verbindet vnd. Komponenten (PC, WLAN, Notebook, ...)

„Rand“ des Internets: „Nutzung“ des Internets

- komponenten: Endsysteme (Client, Server)
 zwischensysteme (Router, WLAN, Mobilfunkzugang)

- Zugangsnetze: Heimnetz, Mobilfunkzugangsnetz, Unternehmensnetz

„Kern des Internets“: Netze untereinander verbundener ISPs, Paketweiterleitung

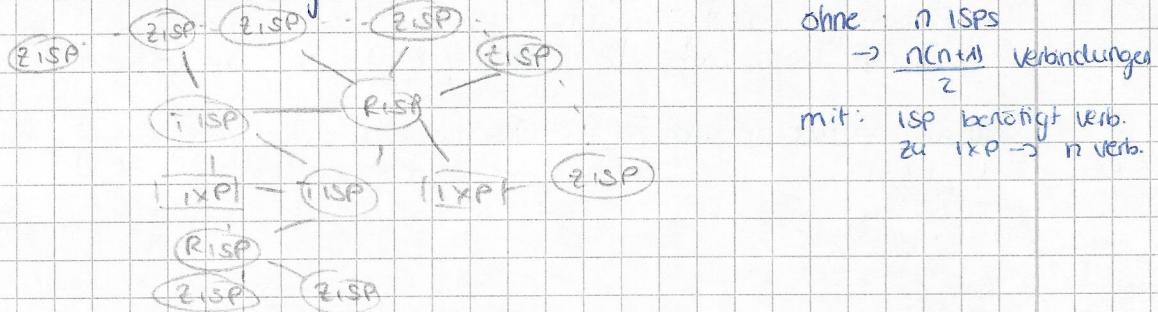
> Internet Service Provider (ISP) = Firma, die Zugang zum Internet zur Verfügung stellt. Verbindung zu anderen ISPs erhält → Erreichbarkeit des Internet

→ Netz-Zugangs-ISPs: Anschluss von Endkunden, Firmen zu ISP

Regionale ISP: Netz - ISP verbinden sich zu regionalen ISPs RISP
 ↳ verbindet sich zu Tier-1-ISP TISP

Tier-1-ISP: Globale Transit - ISPs

IXP (Internet Exchange Point): Daten austausch zw. ISPs IXP



Content Provider = stellt Inhalte bereit → Inhalte möglichst nah zu Kunden bringen

Begriffe

Rechnernetz: besteht aus Komponenten, Übertragungsmedien (verbinden Komponenten)
 PC, Sensor, Handy, ... ↑ Koaxialkabel, Punkt,

Lokales Netz (LAN): > g. Rechnernetz mit geogr. Ausdehnung < wenige km
 > i.d.R. privat (bsp. Unternehmensnetz, Heimnetz)

Weitverkehrsnetz (WAN): Rechnernetz über große geogr. Distanzen
 > privat oder öffentlich (z.B. Telekom Netz)
 Handy, Server, Notebook ↳ Mail, WhatsApp

Endsystem (Host): führt verteilte Anwendungen aus → benötigen Netz zur Kommunikation zw. räumlich verteilten Anwendungsinstanzen
 Router, Switch

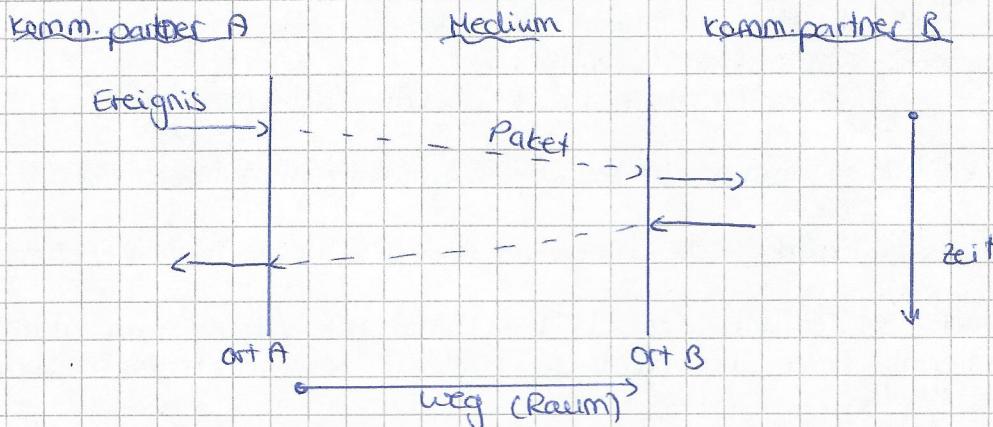
Zwischensysteme: leiten Daten im Netz weiter, führen i.d.R. keine verteilten Anwendungen aus

Datentransmission: Austausch von Daten über (große) Distanzen zw. versch. Parteien
 ↳ Distanzen durch Übertragungsmedien überbrückt
 ↳ Austausch folgt Regeln/Formaten

- (Kommunikations-)Protokoll: definieren Regeln/Formate für Kommunikation zw. Computern, senden/Empfangen von Daten, Ereignissen durchzuführender Aktionen
- > Regeln: zeitl. Ablauf + Aktionen
 - > Format: Syntax + Semantik ausgetauschter Daten

Weg-Zeit-Diagramm: Darstellung räuml. vert. Abläufe

- > vert. Achse: Zeit
- > hor. Achse: räuml. Distanz



(Daten-)Pakete: von Anwendung zu sendende Daten in kleinere Teile gegliedert

- > Nutzdaten: eigentl. Daten
- > Metadaten: für Abwicklung des Protokolls
- ~ Pakete im Netz unabh. behandelt
- ~ durch Netz zum Ziel weitergeleitet (hop-by-hop von zw. sys. zu zw. sys.)

Datenrate (Übertragungsrate): Geschwindigkeit der Übertragung von Daten zw. zwei Komponenten (bit/s)

Grundlegende Prinzipien

Funktionaler Aspekt: "Was" macht ein System (Verben)

Nichtfunktionaler Aspekt: "Wie" macht ein System etwas (Adverbien)

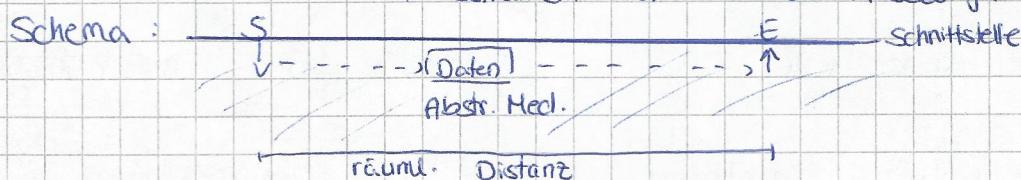
Modell: vereinfachtes Abbild der Wirklichkeit

- > Identifikation + Darstellung wesentlicher Einflussfaktoren
- > Strukturelle Modellbildung: Abstraktion von der inneren Struktur
- > Pragmatische Modellbildung: Interaktion des Systems modellieren

Grundmodell der Kommunikation:

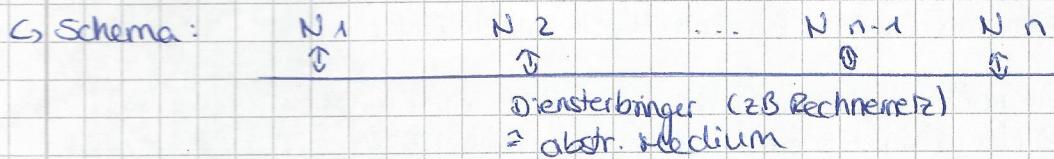
Bestandteile: Sender + Empfänger

Abstraktes Medium: verbindet S + E, stellt Dienst zur Verfügung
↳ abstrahiert von konkreter Umsetzung (-> physikal. Medium)



Dienst

Dienstsicht: Rechnernetz als Block-Box → erbringt Dienst für Netznutzer



Dienst: bündelt zugehörige Funktionen, stellt sie Dienstnehmer zur Verfügung

Dienstzugangspunkt (SAP): Schnittstelle zu Dienst

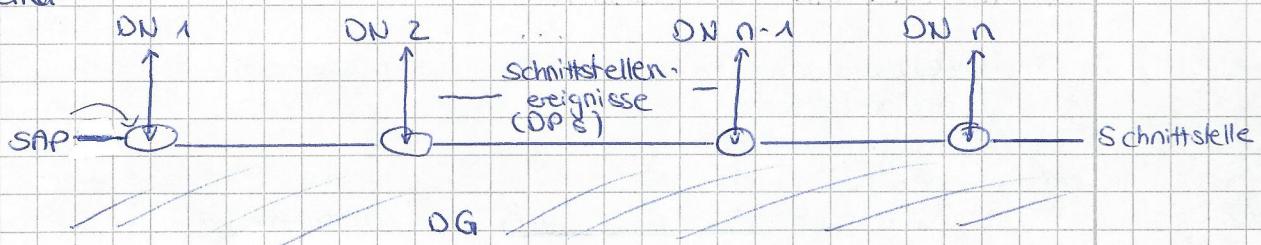
^{ON}
Dienstnehmer: Nutzer, der Dienst in Anspruch nimmt

^{DG}
Dienstgeber: stellt Dienst zur Verfügung

^{DP}
Dienstprimitive: beschreibt Interaktion zw. Dienstnehmer und -geber

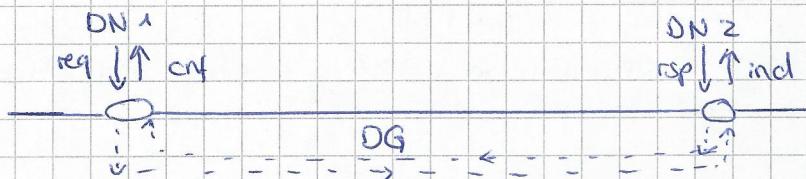
Beschreibung eines Dienstes reflektiert Verhalten an SAPs zu Dienstgeber

Schemat:



DP Typen am SAP:

- > Request: Beauftragung ($DN \rightarrow DG$)
- > Indication: Benachrichtigung des Partners ($DG \rightarrow DN$)
- > Response: Beantwortung durch Partner ($DN \rightarrow DG$)
- > Confirmation: Benachrichtigung über Abschluss ($DG \rightarrow DN$)



Unbestätigter Dienst:

nur Req + Ind, Initiator erhält keine Rückmeldung vom Beantworter

Bestätigter Dienst:

Req + Ind + Resp + Cnf, Initiator erhält Antwort vom Beantworter

Zuverlässiger Dienst: am SAP des Empfängers gilt:

- alle empfangenen Daten sind korrekt
- alle gesendeten Daten vollständig und in richtiger Reihenfolge
- keine Duplikate
- keine Phantom-Daten

unzuverlässiger Dienst: keine Aussage über Korrektheit, Reihefolge, Duplikate

! Bestätigt + Zuverlässig!

Verfeinerung des Grundmodells

Abstraktion auf Basis von Schichten

- ↳ Bereitstellen von Diensten an Schnittstelle nach oben
- ↳ Nutzen von Diensten an Schnittstelle nach unten

Schicht:

- bündelt zusammengehörende Funktionalitäten
- bietet wohldefinierte Dienste an Schnittstelle

→ Schichten hierarchisch angeordnet, evtl. in Teilschichten gegliedert

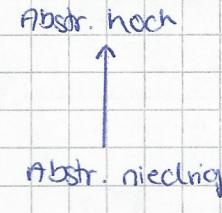
→ Dienst einer Schicht durch Zusammenwirken der Protokollinstanzen

Horizontale Kommunikation: zw. S+E einer Schicht, Protokollinstanzen einer Schicht tauschen Daten um Dienst zu erbringen

Vertikale Kommunikation: zw. Schichten innerhalb eines Geräts
Interaktion zw. Protokollinstanzen der Schichten N-1, N, N+1

Referenzmodell mit 5 Schichten

- 5 Anwendungsschicht
 - 4 Transportschicht
 - 3 Vermittlungsschicht
 - 2 Sicherungsschicht
 - 1 Physikalische Schicht
- übertr. medium = ÜM



Physikalische Schicht:

Zwei direkt verbundene Geräte (über ÜM) können Bits austauschen

PS ← Bits → PS

| Gerät 1 | ÜM | Gerät 2 |

- Übertragung von unstrukturierten Bitfolgen über phys. Medium
- Bietet S2 unzuverlässigen Dienst

Sicherungsschicht:

- Datenaustausch zw. phys. benachbarten Geräten
- Pakete hier: Rahmen
- Bereitstellung zuverlässiger und unzuverlässiger Dienste
- Adressierung der Geräte
- Pufferung der Daten bei S+E
- Strukturierung der Übertragung im Rahmen

Vermittlungsschicht:

- Datenaustausch zw. nicht direkt benachbarten Geräten
- Pakete hier: Datagramme
- Verknüpft Teilstücke zu Ende-zu-Ende-Strecken
- Wegwahl
- Addressierung der Endsysteme
- Bereitstellung zuv. und unz. Dienste
- Datagramme können in zw. Systemen gepuffert werden
- Datagramme in Rahmen der Sicherungsschicht übertragen

Transportschicht

- Datenaustausch zw. Anwendungsprozessen
- Pakete hier: Segmente / Datagramme
- Bereitstellung zw. und zw. Dienste
- Pufferung der Daten im Endsystem
- Adr. von Anwendungsprozessen
- Segmente / Datagramme in Datagramm(e) der Verm. Schicht übertragen

Anwendungsschicht:

- Anwendung + erforderliche Mechanismen, Datenstrukturen
- Pakete hier: Nachrichten

Nutzer-zu-Nutzer:

- Transportschicht
- logische Komm. zw. Prozessen

Ende-zu-Ende:

- Vermittlungsschicht
- logische Komm. zw. Endsystemen

TCP/IP-Referenzmodell

4 Schichten:

Anw. Schicht
Transportschicht
Internetschicht
Netzzugang

→ ≈ phys. Sch. + Sch. Schicht

OSI-Referenzmodell

- 7 Schichten:
- 7 Anw. Schicht
 - 6 Darstellungssch.
 - 5 Sitzungssch.
 - 4 Transportsch.
 - 3 Vermittlungssch.
 - 2 Sicherungssch.
 - 1 Physische Sch.

Anwendungsoorientierte Schichten

Transportorientierte Schichten

Transportorientierte Schichten:

Datenübertragung zw. Anwendungssystemen

- ↳ Semantik der Daten unabhängig von Funktionsweise des Transports
- ↳ nur Bedürfnissen des Datenaustauschs, bereitstellenden Dienstes unterstellt

Anwendungsoorientierte Schichten:

Anwendung bezogene Aspekte

- ↳ Semantik der Daten wichtig → Inform. darstellung + - austausch

Sitzungsschicht: bietet Wichtunterbrechbarkeit der Komm. Beziehung
Gliederung des Datenaustauschs bzgl. der Anwendung

Darstellungsschicht: einheitl. Darstellung der Daten, Beibehaltung der Semantik der Information

Anwendungsschicht: Austausch → anwendungsabh. Daten

Physikalische Schicht

Abstraktes Modell nach Shannon: allg. Erkenntnisse, die für alle Arten der Datenübertragung gelten



- > Informationsquelle: liefert Bitfolge in vorgegebenem Takt an Sender
- > Sender: wandelt Nachricht in Signal um, sendet dieses auf den Kanal (Medium hier: (Übertragungs-)Kanal)
- > Empfänger: rekonstruiert Bitfolge aus Empfangssignal

Medium \rightarrow Modell: Kanal

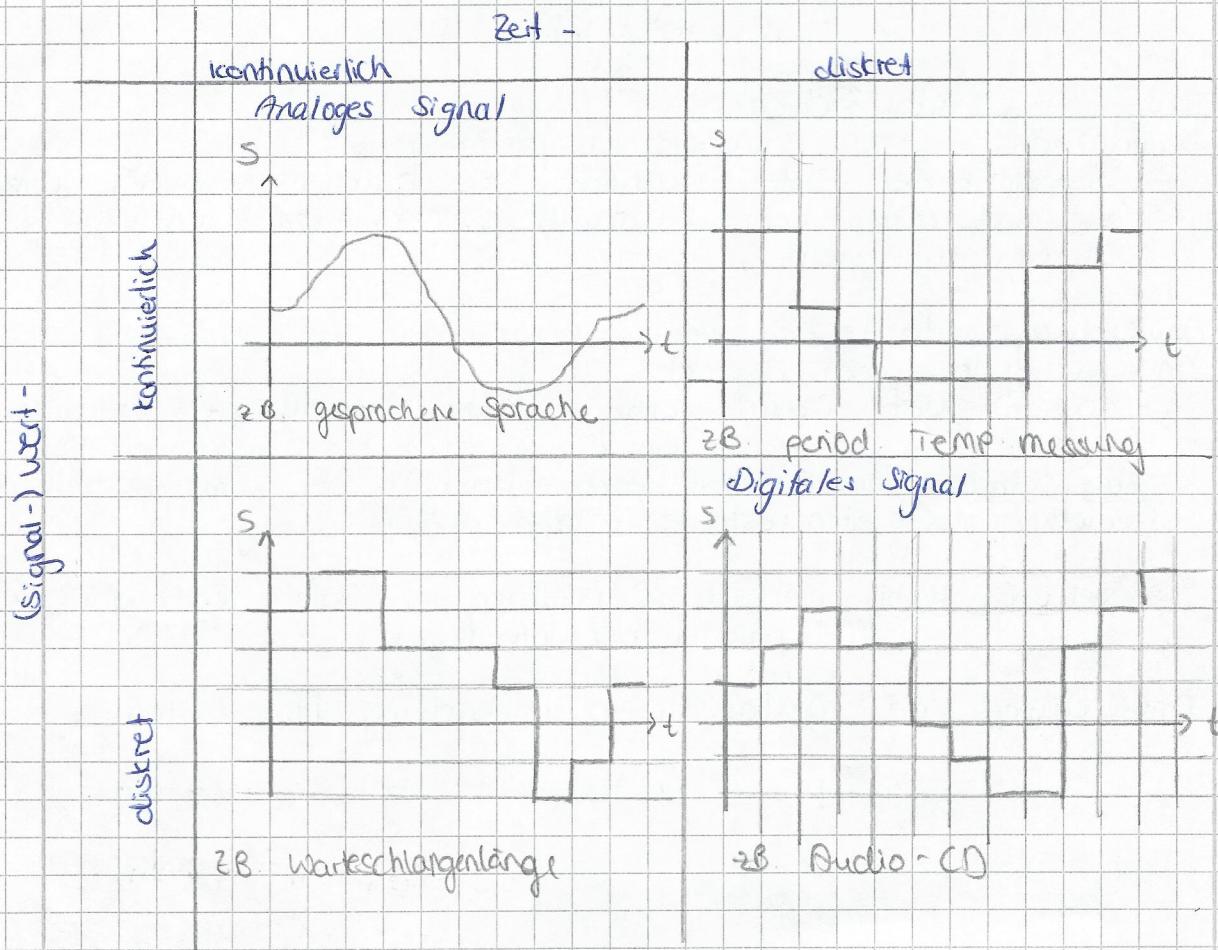
Aufgabe: physik. Verbindung benachbarter Geräte

\rightarrow über überbrückbare Distanz, "Verlängerung" durch Repeater (\rightarrow Signalverstärker)

Signal

Signal = math. Funktion von mind. einer unabh. Variablen.
Signal ändert Wert als Funktion der Zeit $\rightarrow s(t)$

Klassifizierung:



Digitales Signal: zeit- und wert diskret

→ isochrones digitales Signal: Wechselt das Signalwerts nur zum Schritttakt (feste Schrittdauer)

Analoges Signal: zeit- und wert kontinuierlich

→ periodisches Signal: periodendauer T : $s(t+T) = s(t)$ $-\infty < t < \infty$
Frequenz $f = \frac{1}{T}$

Fourier-Reihe: period. Fkt als Summe von sin/cos-Kurven unterschiedl. Frequenzen

$$\text{per. Fkt. } x(t) = \frac{A_0}{2} + \sum_{n=1}^{\infty} [A_n \cos(2\pi n \cdot f_0 t) + B_n \sin(2\pi n \cdot f_0 t)]$$

$$A_0 = \frac{2}{T} \int_0^T x(t) dt, \quad A_n = \frac{2}{T} \int_0^T x(t) \cos(2\pi n f_0 t) dt$$

$$B_n = \frac{2}{T} \int_0^T x(t) \sin(2\pi n f_0 t) dt$$

f_0 = Kehrwert der Periode von $x(t)$

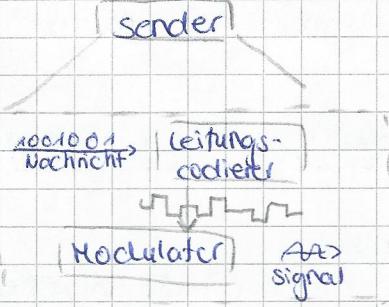
Amplitude: Höhe des Ausschlags

Frequenz: "Häufigkeit" / Zeit

Phase: Verschiebung

Modulation

- > Leistungscodierer: bereitet Bitstrom auf Übertragung vor
- > Modulator: Bildung phys. Signale



Modulation: Aufbereitung eines zu übertragenden Nutzsignals durch Modulation sinusförmiger Trägersignale für phys. Übertragung

- > Amplitudenumtastung: Veränderung der Amplitude des Trägersignals
 - > Frequenzumtastung: Umschalten zw. versch. Trägerfrequenzen
 - > Phasenumtastung: Umschalten durch Phasensprung
- bei normaler Amp. linear
bei doppelter Amp. doppelte Frequenz
Schwingung

Kanalkapazität

phys. Medien übertragen nur endliches Frequenzband
↔ festlegen von oberer Grenzfrequenz f_u und unterer fu

Bandbreite: Breite des Spektrums der involvierten Frequenzen $B = f_u - f_l$

Kanalkapazität: $C = B \cdot \log_2 \left(1 + \frac{S}{N} \right)$

Bit/s Bandbreite in Hz (1/s) Energie des Signals
Datenrate Anzahl Signalstufen Energie der Störquelle

$\text{SNR} [\text{dB}] = 10 \log_{10} \frac{\text{Sign.-energie}}{\text{Störenergie}}$

Signal Noise Ratio

max. Datenrate, die unter geogr. phys. Bedingungen über längeren Zeitraum aufrecht erhalten werden kann

Leistungscodierung

Codierung: Σ Quellenalphabet, Π Codealphabet (je endlich)

\rightarrow Codierung $c: \Sigma^+ \rightarrow \Pi^+$ injective Abbildung

Code: c Codierung, Code = Menge $C = c(\Sigma) = \{c(\sigma) | \sigma \in \Sigma\}$
 $c(\sigma)$ = Codewort von σ

binäres digitales Signal: zeit-, wertdiskret, kann nur 0 oder 1 sein

mehrwertiges dig. Signal: zeit-, wertdiskret, kann mehr als 2 Werte annehmen

Datenrate = Übertragungsgeschw.: Anzahl übertragbarer Bits / Schrittdauer
 \rightarrow bit/s

Baudrate = Schrittggeschw.: Zahl der Signallwert-Zustandswechsel / Schrittdauer
 \rightarrow baud (1/s)

• isochr. Signale: $1/T$

(Datenrate bei Nachricht relevant, Baudrate zw. Leit. cod. und. Mod.)

Leistungscodes: Eigenschaften

- > Taktrückgewinnung: sender, Empf. ; dR kein gemeinsamer Takt
 \hookrightarrow Empf. sollte aus Signalwerten Takt rekonstruieren können
- > Geringer Gleichstromanteil: Running digital sum ≈ 0 nicht gg. oo
 \hookrightarrow meist nur im statistischen Mittel erfüllbar
- > Niedrige Baudrate
- > Geringe Komplexität (= geringe Kosten, Baugröße, ...)

Von-Return-to-Zero Code

- bin. Code \rightarrow Symbolwerte durch Signalwerte bestimmt

1: hoher Pegel über gesamtes Taktintervall

0: niedriger Pegel - " -

Signalwechsel an Intervallgrenzen

Signalerkennung durch Pegel

⊕ Baudrate ⊕ Taktrückgew. bei langen 0-11-Folgen nicht möglich
Gleichstromanteil RDS $\rightarrow \infty$

Manchester Code

- biphasen Code: Symbolwerte durch Phasensprünge bestimmt

1: Signalübergang vom hohen zum niedrigen Pegel in Intervallmitte

0: niedr. hohen

Hilfswechsel bei 0ern / 1ern in Folge

Signalerkennung durch Signalwechsel in Intervallmitte

⊕ Taktrückgew., keine Gleichstromkomponente, Komplexität gering

⊖ Baudrate

AMI Code

- ternärer Code: Symbolwerte 0, 1 auf drei Signalwerte abgebildet
 - 1: abwechselnd durch pos. oder neg. Pegel über ges. Intervall
 - 0: Null-Pegel
- Signalerkennung durch Signalwert im Intervall

⊕ Gleichstromfreiheit, Baudrate

⊖ Taktrückgew. bei langen 0-Folgen nicht möglich

Störquellen

Dämpfung

Signalstärke sinkt mit zunehmender Distanz

- ↪ Signalstärke beim Empf. muss groß genug sein um Signal zu erkennen
 - ↪ muss auch größer als Rauschen sein
- Einsatz von Verstärkern / Repeatern

Verzögerungsverzerrung

durch Signalausbreitungsverzerrung

- ↪ variiert mit Frequenz → Phasenverschiebung zw. Frequenzen

- ↪ Überlappung aufeinander folgender Bits → Intersymbol-Interferenz

Rauschen

→ Thermisches Rauschen: Flkt. der Temperatur, gleichmäßig verteilt über Frequenzen, nicht eliminierbar

→ Modulationsräuschen: zwei Frequenzen f_1, f_2 überlagern sich $\Rightarrow f = f_1 + f_2$

→ Impulsrauschen: unregelmäßige Ausschläge, Geräuschspitzen kurzer Dauer und hoher Amplitude
 ↪ ausgelöst durch elektromagn. Störungen (z.B. Blitze)

Sicherungsschicht

Datenaustausch zw. physikalisch benachbarten Geräten

- ↪ Pakete = Rahmen

Basiert auf Dienst der physikalischen Schicht

implementiert in End- & Zwischensystemen

- ↪ auf Netzinterface oder Chip

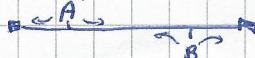
- ↪ an Gerätbus angeschlossen

Eigenschaften

- Bereitstellung zuverlässiger und unzuverlässiger Dienste
 - ↪ für zuverlässig: Erkennen + Beheben von Bitfehlern notwendig
- Adressierung der Geräte
- Dateneiffernung
- Strukturierung der Rahmenübertragung
- Abstrahierung vom physik. Medium

Broadcast-Link

alle an Broadcast-Medium angeschlossenen Geräte können alle gesendeten Rahmen "sehen"



Punkt-zu-Punkt-Link

zwei Geräte über dedizierten Link verbunden A - B

Punkt-zu-Punkt-Kommunikationsformen

- > Simplex: Übertragung nur in eine Richtung \rightarrow
- > Halbduplex: Übertragung in beide Richtungen, abwechselnd nicht zeitgleich \Rightarrow
- > Duplex: Übertragung in beide Richtungen, zeitgleich \leftrightarrow

Bitfehler

Verfälschung von Bits während Übertragung

Einzelbitfehler = einzelnes Bit fehlerhaft

Schlüsseelfehler = mehrere direkt aufeinanderfolgende Bits fehlerhaft

Synchronisationsfehler = alle Bits ab dem Fehler werden falsch interpretiert

Bitfehlerrate Maß für Fehlerhäufigkeit = $\frac{\Sigma \text{ gestörte Bits}}{\Sigma \text{ übertragene Bits}}$

Fehlerauswirkung Auswirkung einer Störung, u.a. abh. von Datenrate
 $N \text{ Bitfehler} \quad N = \text{Datenrate (bit/s)} \cdot \text{Störungsdauer (s)}$

Paketfehler

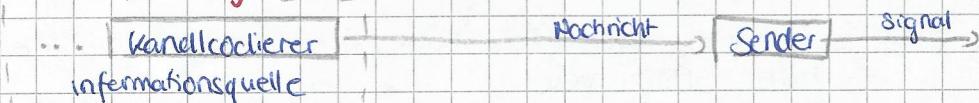
- > Verlust eines Pakets
- > Duplizierung eines Pakets

- > Empfang Phantom-Paket
- > Reihenfolgeverfälschung

Redundanz

- > Fehlererkennung (Error Detecting Code EDC)
 - Datenredundanz
 - Codewörter verwenden, die sich hinreichend unterscheiden
- > Fehlerkorrektur (forward Error correction, FEC)
- > Wiederholungsaufforderung (Automatic Repeat Request ARQ)

Kanalcodierung



Kanalcodierer schützt Daten durch Hinzufügen von Redundanz gegen Übertragungsfehler

Erkennung von Bitfehlern

Paritybits Codewort $z = (xp)$ $x = \text{Nutzdaten} = x_1 x_2 x_3 \dots x_k$

$$p = f(x) = x_1 \oplus x_2 \oplus \dots \oplus x_k$$

\rightarrow nur einzelne Fehler erkennbar

r-fehlererkennender Code Empfänger erkennt Codewort als fehlerhaft indem bis zu r Symbole verfälscht

r-fehlerkorrigierender Code Empfänger kann Codewort indem bis zu r Symbole verfälscht wurden wieder herstellen

Hammig-Abstand C ist Code fest Länge, v_1, v_2 Codewörter aus C
 $\Delta(v_1, v_2) = \# \text{anzahl untersch. Positionen}$

Code-Abstand minimale Distanz zw. zwei Codewörtern
 $\Delta C = \min \{ \Delta(v_1, v_2) \mid v_1, v_2 \in C, v_1 \neq v_2 \}$

MASCHMEYER GROUP



Hamming-Kugel $C \subseteq \mathbb{F}^n$ Code fester Länge, $v \in C$, $r \in \mathbb{N}$

$K_r(v) = \{w \in C \mid \|v, w\| \leq r\}$ = Hamming-Kugel von Wort v mit Radius r
 ↳ alle Codewörter die sich von v in höchstens r Pos. unterscheiden

$\Rightarrow C$ ist r -fehlererkennend $\Leftrightarrow \Delta C > r$
 C ist r -fehlerkorrigierend $\Leftrightarrow \Delta C > 2r$

C Code mit ΔC ungerade, C' um ein Paritätsbit erweiterter Code:
 $\Delta C' = \Delta C + 1$

Hamming-Code linearer (n, k) Code $n = \text{länge Codewort}, k = \text{Anzahl Nutzdatenbits}$
 Hamming-Abstand von 3
 ↳ kann Einzelbitfehler korrigieren

Linearer Code: \mathbb{F}^n Vektorraum über endl. Körper \mathbb{F} , $C \subseteq \mathbb{F}^n$ = linearer (n, k) -Code
 falls C k -dim. UVR von \mathbb{F}^n

→ Hammingcode hat Paritätsbit an allen Zer-Potenz-Stellen

$$z = (p_1 p_2 x_1 p_3 x_2 x_3 x_4)$$

$$\begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix}$$

		p_4	p_3	p_2	p_1
Pos in z	Code	2^3	2^2	2^1	2^0
		3	5	6	

Syndrom Berechne XOR aller Stellen die in Par. bit einfließen + Par. bit selbst

$$\text{zB: } s_1 = x_1^* \oplus x_3^* \oplus x_4^* \oplus p_1^*$$

$\Rightarrow s_3 s_2 s_1 \dots$ ist Syndrom $\rightarrow s = (0, 0, 0) \rightarrow$ fehlerfreie Übertragung

Hamming-Schranke Wie viele Prüfbits erforderlich um r Fehler korrigieren zu können?

bei perfektem Code: alle Empfangswörter innerhalb Korrigierkugeln

$$\Rightarrow 2^n \geq 2^k \cdot \sum_{i=0}^r \binom{n}{i} \quad (?)$$

Cyclic Redundancy Check CRC

Linearer Code ist zyklisch, falls jede zyklische Verschiebung eines Codeworts ebenfalls ein Codewort ist

Polynome Symbole eines Codeworts als Koeff. eines Polynoms z.B. $0101 = x^2 + 1$

Generatorpolynom $g(x) \in \mathbb{F}[x], n \in \mathbb{N}^+$ $g(x)$ generiert Code der Länge n :
 $C = \{v(x) \mid \deg v(x) < n, g(x) \text{ teilt } v(x) \text{ ohne Rest}\}$

↪ $g(x)$ erzeugt genau dann zyklischen Code der Länge $n \Leftrightarrow g(x) \text{ teilt } x^n - 1 \text{ ohne Rest}$

CRC-Algo Sender geg. $g(x)$, $\deg g(x) = r$

Nutzdaten $u(x)$, $\deg u(x) < n-r \rightarrow$ an Nutzdaten r Oen anhängen

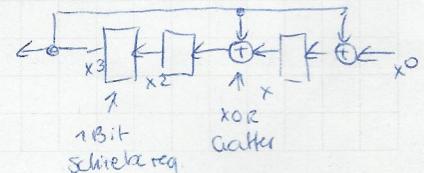
ges. Codewort $v(x)$, $\deg(v(x)) < n$

$$(1) \quad u(x) \cdot x^r \quad \checkmark \text{ mod 2 division} \rightarrow \text{Polynomdiv}$$

$$(2) \quad x^r \cdot u(x) / g(x) = r(x)$$

$$(3) \quad r(x) + x^r u(x)$$

schieberregister $g(x) = x^3 + x + 1$



Empfänger geg. $g(x)$, $v(x)$ ges. $u(x)$

$$(1) \quad v(x) / g(x)$$

(2) Kein Rest \rightarrow Kein Fehler festgestellt
 Rest \rightarrow fehlerhafte Übertragung

Erkennen von Bitfehlern

- alle Einzelbitfehler: x^r, x^e des Generatorpolynoms dürfen nicht 0 sein
- nahezu alle Doppelbitfehler: $g(x)$ mind. 3 Terme, (x^{k+1}) nicht durch $g(x)$ teilbar
- ungerade Anzahl Bitfehler: $g(x)$ muss Faktor $x+1$ enthalten
- alle Bursts mit bis zu m Bitfehlern: $g(x)$ hat Grad m

Vorwärtsfehlerkorrektur FEC

Sender sendet redundante Pakete \rightarrow Empfänger kann Paketfehler korrigieren ohne Sendewahl.

- ① Reduziert Verzögerung bis Daten beim Dienstnutzer abgeliefert werden können
- ② Redundante Info unabh. von evtl. auftretendem Fehler gesendet

Automatic Repeat Request ARQ

Verfahren zur Sendewiederholung bei Paketfehlern

Fehlererkennung, -behebung

Sequenznummern: Kennung je Paket

Quittungen: Empfänger informiert Sender ob er Paket empfangen hat oder nicht

↳ ACK: Positive Quittung

NACK: Negative Quittung

SACK: Paket wurde korrekt empfangen

Kumulativ: Quittung für eine Menge von Paketen

Zeitgeber: abh. von zeitl. Obergrenze: Vermutung, dass Paket nicht angekommen

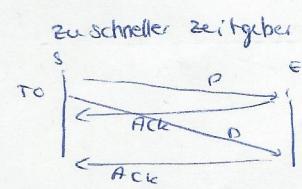
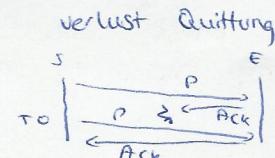
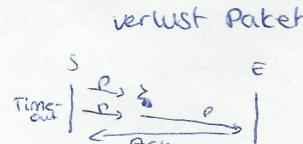
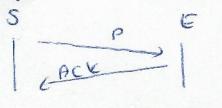
↳ Sender weilt.

Stop-and-Wait

Sender sendet Paket, wartet auf Quittung bevor neues Paket \rightarrow Wdh. falls keine Quittung

Szenarien

Regulärer Ablauf



- ↳ verwendet Sequenznummern um doppelte Pakete untersch. zu können
- ↳ 0,1 als Nummern im Wechsel ausreichend

Alternating Bit Protocol ABP

Paket: |S| Nutzdaten |CRC| S-Sequenznr. 1Bit

Sendet Paket, warte auf Quittung für zuletzt gesendetes Paket

↳ Quittung nur für korrekte Pakete (richtiges S, CRC)

Leistungsbewertung

Durchsatz Datenrate Menge an Daten, die pro Zeiteinheit übertragen werden

Auslastung Verhältnis tats. Nutzung zu mögl. Nutzung

Sendezzeit Zeit um n Bits zu senden $t_s[s] = \frac{x[\text{bit}]}{r[\text{bit/s}]} = \frac{\text{zu sendende Datenmenge}}{\text{zur Verfügung stehende Datenrate}}$

Ausbreitungsverzögerung Zeit die ein Bit von A nach B benötigt

$$t_a = \frac{d[m]}{v[m/s]} = \frac{\text{Distanz}}{\text{Länge Medium}} \cdot \frac{1}{\text{Ausbreitungsgeschw. Signal}}$$

Verarbeitungsverzögerung Zeit die nötig um Paket zu verarbeiten

Warteschlangenverzögerung Zeit die gewartet werden muss bis Paket geordnet werden kann

MASCHMEYER GROUP



$$t_s = \frac{x}{r} \quad \begin{array}{l} x = \text{Datenmenge} \\ r = \text{Datenrate} \end{array}$$

$$t_a = \frac{m}{v} \quad \begin{array}{l} m = \text{Länge Med.} \\ v = \frac{\text{Datenrate}}{\text{Geschw.}} \end{array}$$

Stop-and-Wait-Analyse

Gesamtübertragungszeit pro Paket: $t_{\text{ges}} = t_s(\text{Daten}) + t_a + t_u(\text{Daten}) + t_s(\text{Quittung}) + t_a + t_u(\text{Quittung})$

$$\text{Auslastung } U = \frac{t_s}{t_{\text{ges}}} = \frac{t_s}{t_s + 2t_a} = \frac{1}{1+2a} \quad a = \frac{t_a}{t_s} = \frac{\frac{m}{v}}{x} \rightarrow \begin{array}{l} \text{länge Medium} \\ \text{Datenrate} \end{array}$$

$\rightarrow \text{Bandbreiten-Verzögerungs-Produkt}$

$\hookrightarrow \text{Paketläng.} = \text{länge des Mediums in Bit}$

\rightarrow bei Übertragungsfehlern: $p = \text{Wkt.}, \text{ dass Paket fehlerhaft}$

$$\hookrightarrow U = \frac{1-p}{1+2a}$$

Go-Back-N

Sender: sendet max. N Pakete bis Quittung \rightarrow Sendefenster: Menge der Daten, die sender noch senden darf bevor er Quittung erhalten muss $\rightarrow SF=0$: keine Daten außer sendewahl.

Empfänger: quittiert i.d.R. mit kumulativen Quittungen

Fehlerfall: Empfänger: Empfang fehlerhaftes Paket / falsche Reihenfolge \rightarrow Paket + nachfolgende verwerfen

Sender: Ablauf des entspr. Zeitgebers \rightarrow Wdh. aller noch nicht quittierten Pakete

Sendepuffer: Sender puffert Pakete die evtl. wiederholt gesendet werden müssen

Empfangspuffer: Empfänger puffert Pakete, falls nicht direkt an überliegende Schicht weitergabbar

Analyse: W Sendefenstergröße, $W \geq 1+2a$: ohne Pause senden \rightarrow 100% Auslastung
 $W < 1+2a$: kein senden nach Aufbrauchen von SF

$$U = \begin{cases} 1, & W \geq 1+2a \\ \frac{W}{1+2a}, & \text{sonst} \end{cases} \quad (\text{ohne Fehler})$$

mit Fehler:

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

Selective Repeat

Sender: wie bei Go-Back-N

Empfänger: quittiert mit selektiven Quittungen

Fehlerfall: Empfänger: Nachfolgende, korrekte Pakete in Empfangspuffer + bestätigen

Sender: nur nicht korrekt empfangene Pakete wiederholen

Selective Reject

Empfänger teilt mit negativer Quittung SREJ (Sequenznr.) mit, dass Paket nicht korrekt empf.
 Sender wiederholt Paket sofort, ohne Timeout

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

mit Übertragungsfehlern

Flusskontrolle

Problem: sender kann Empfänger überlasten \rightarrow Datenverlust da Puffer nicht ausreichend

open Loop: Beschreibung des Verkehrs mit anschl. Ressourcenreservierung + Überwachung eingetekter Verkehr

Closed Loop: Rückkopplung um Problem zu verhindern \rightarrow Sender adaptiert Datenstrom entsprechend

Halt-und-Weiter

Empfänger sendet HALT: kann nicht mehr mit Sender schritt halten

WEITER: nach HALT wieder in der Lage Daten zu empfangen

\rightarrow nur auf Vollduplex-links verwendbar, bei hohen Verzögerungen nicht effektiv, Verlust HALT?

Kreditbasierte Flusskontrolle

Send-Kredit: max. Anzahl Pakete die Sender ohne Quittung senden kann $\hat{=}$ Pufferkapazität Empf.

Sliding Window: Sender, Empfänger verwalten Send-/Empfangsfenster SF/EF
es gilt: $SF \geq 0 \wedge EF \geq 0$

Ann.: Fenstergr. in Anzahl Pakete N, pos. selekt. Quittungen

Sender: $(SF > 0) \wedge (N > 0)$: Sender Paket, $SF = SF - 1$

Quittung empfangen: $SF = SF + 1$

Empfänger: korrektes Paket in Reihenfolge: $EF = EF - 1$

Quittung gesendet: $EF = EF + 1$

Verbindungen

(kommunikations-) Verbindung = komm. bez. zw. 2 oder mehr Komm. partnern \rightarrow durch gemeinsamen (Verbindungs-) Kontext charakterisiert

\hookrightarrow Basis für Bereitstellung zuverl. Dienste

Verbindungsorientierte Kommunikation

Vor Datenaustausch: Verbindungsaufbau, danach: Verbindungsabbau

↓ etablieren Verbindungskontext ↓ Kontext lösen, Ressourcen freigeben

Verbindunglose Komm.

Kein Kontext zw. komm. partnern \rightarrow Daten sofort versenden

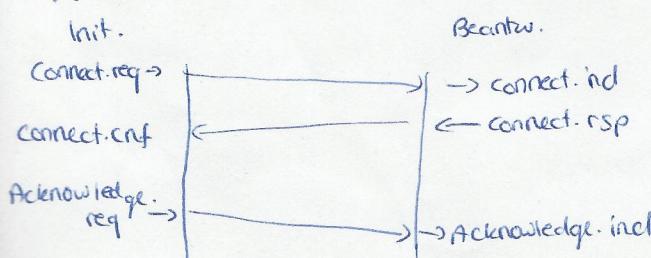
Verbindungsauftbau: 2-weg-Handelshalke

erfolgreicher Aufbau Initiator Beantworter
connect. req. \rightarrow \leftarrow connect. ind.
connect. conf. \leftarrow \leftarrow connect. rsp

Mög. Fehler: Abort nach connect. ind

connect. rsp nicht empfangen

3-weg-Handelshalke



Verbindungsabbau

zB geregelt: Partner kann Abbau zustimmen / ablehnen

Verbindungsabbruch

Außerplanmäßiger Verbindungsabbau

Erbringerabbr.: kann immer passieren, keine Gleichzeitigkeit, bestimmte Reihenfolge garantiert

Nutzerabbr.: Verbindung gilt für Initiator als sofort abgebrochen

Netztopologien

Vollvermaschtes Netz

jedes Gerät über separaten physik. Link mit jedem anderen Gerät verbunden
Gefordert $\frac{N(N-1)}{2}$ Links bei N Geräten \rightarrow skaliert nicht



Teilvermaschtes Netz

nur ein Teil der Geräte direkt verbunden \rightarrow Kommunikation zw. nicht verbundenen Geräten über andere Geräte



Hierarchische Topologie/ Baum



Stern

zentrale Komponente, zu der jedes Gerät physikalischen Link hat
 \hookrightarrow single-point-of-failure

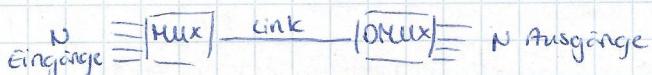


Ring

Linksausfall: eingeschränkte Kommunikation

Multiplexverfahren

Multiplexen = simultane Nutzung eines physikalischen Links durch mehrere Signale



Bus

Linksausfall: keine Kommunikation

in mehrere Dimensionen möglich: Raum, Zeit, Frequenz, Code

Raummultiplex Space Division Multiple Access SDMA

Signal bekommt Link exklusiv zur Verfügung gestellt

> drahtgebunden: ein physik. Link pro Übertragung

> drahtlos: gleiche Frequenz mit ausreichendem räuml. Abstand z.B. Mobilfunknetze Zellen

Frequenzmultiplex Frequency Division Multiple Access FDMA

Bandbreite in untersch. Frequenzbänder aufteilen, je ein Signal zuordnen



\rightarrow Zeitmultiplex Time Division Multiple Access TDMA

Aufteilung in Zeitschritte in denen ein Signal ges. Bandbreite nutzen kann



zuteilung von Zeitschritten:

> statisch: offline konfiguriert, keine Änderung während Betrieb

> dynamisch: - fest: Signalisierungsprotokolle während Betrieb

- variabel: Zugriff on-demand, z.B. Ethernet, WLAN

Codemultiplex Code Division Multiple Access CDMA

Signal wird vom Sender mit für ihn eindeutigen Pseudorandom-Zeichenfolge codiert, kann vom Empfänger wiederhergestellt werden, alle Sender senden gleichzeitig im gleichen Frequenzkanal

Medienzuteilung

Häufiges Kommunikationsmuster in der Datenkommunikation: on-off

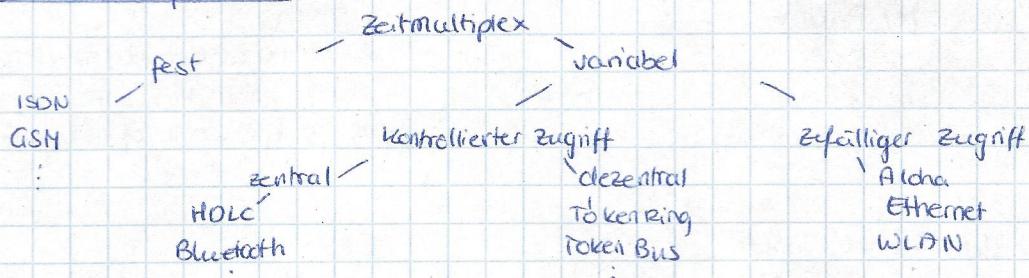
on: Daten senden mit höchster verfügbarer Datenrate

off: keine Daten senden

\rightarrow toff \gg ton, unklar zu welchem Zeitpunkt Kommunikation beginnt

lokale Netze: Verfahren zur variablen on-demand Medienzuteilung: MAC = Media Access Control

Kategorien + Beispielnetze



Aloha erstes MAC Protokoll für paketbasierte drahtlose Netze.

- Medienzuteilung: Zeitmultiplex, variabel, zufälliger Zugriff

- keine Ankündigung des Sendewunsches / keine vorherige Medienüberprüfung + asynchroner Zugriff

=> Kollisionen möglich max. Auslastung > 18,4%

Kollisionserkennung

Explizit: Bestätigung durch höhere Schichten \rightarrow separater Kommunikationskanal erforderlich

Implizit: bei Nutzung von Aloha über Satelliten: Sendezeit broadcastet empfangene Rahmen an alle, sendende station empfängt Rahmen nach RTT wieder \rightarrow auf Korrektheit prüfen

Reaktion: Sendewechl. zu randomisiertem Zeitpunkt, Abbruch nach max. mtz. gescheiterten Sendewechl.

Slotted Aloha $\hat{=}$ synchronem Aloha \rightarrow Zugriff nur zu Beginn eines Zeitschlitzes

\hookrightarrow im Mittel weniger Kollisionen als Aloha max. Auslastung = 36,8%.

CSMA (Carrier sense Multiple Access, zufälliger Zugriff)

\rightarrow Listen before talk: Gerät prüft vor Senden, ob Medium frei ist

\hookrightarrow sendet erst wenn frei, aber mehrere Geräte können gleichzeitig beginnen zu senden \rightarrow Kollision

CSMA/CD: CSMA with Collision Detection $t_s > 2 \tau_a$

\rightarrow Listen while talk: Kollisionserkennung durch Abhören während des Sendens

\rightarrow Kollisionsfall: senden abbrechen, später erneut versuchen

CSMA/CA: CSMA with Collision Avoidance

\rightarrow sendende Station schließt auf Kollision bei Ausbleiben einer Quittung

"Zeitfenster" für potentielle Kollisionen

maximale Ausbreitungsverzögerung τ_a , max zw. 2 an Medium angeschl. Geräten

Annahme CSMA: $a = \frac{t_s}{\tau_a}$ klein

Zufälliger Zugriff Umsetzung bei Ethernet

- Kollision: Abbruch Sendekreislauf, Senden von Jamming-Signal, wdh. Senden regelt Backoff-Algo

- Voraussetzungen zur Kollisionserkennung:

Mindestlänge der Rahmen: Senden darf nach Signallaufzeit durch Medium + zurück noch nicht beendet sein

\hookrightarrow Padding falls zu kleiner Rahmen

p-Persistenz

Medium frei: sende mit Wkt $0 < p \leq 1$
 \hookrightarrow in Stationen warten \rightarrow setze $p < 1/n$

Exponentieller Backoff sendewahl nicht erfolgreich \rightarrow vermeide erneute Kollision
 Warte zufällige Zeit aus Zeitintervall bis Start Sendewahl.
 \hookrightarrow verdopple Intervall je erfolgloser Wdh.

Token passing Token = spezieller Rahmen zur Vergabe des Senderechts
 → Weitergabe von Station zu Station, Station die Token besitzt darf senden
 → Regeln zur Weitergabe!

Token Ring Stationen als Punkt-zu-Punkt-Ring, unidirektional
 Station mit Token darf senden, Paket kommt wieder bei dieser Station an (Ring)
 \hookrightarrow gebe dann Token an nachfolgende Station

max. Sendezzeit begrenzt: Token Holding Time, Kontrollinfo per piggybacking zum Sender,
 Prioritätsmechanismus verfügbar

Monitor ausgewählte Station zur Funktionsfähigkeitsüberwachung

→ meist nicht als Ring mit direkten Links, sondern als Stern
 \hookrightarrow Ringverteiler in der Mitte
 ⇒ Strukturierte Verkabelung von Gebäuden möglich, zuverlässig

Token Bus Verbindung der Stationen durch Bus \rightarrow Bildung logischer Ring zur Token Weiterleitung

Token wird über Bus weitergegeben, nur logischer Nachfolger darf Token entgegennehmen
 → Ablösche auf Bus auf Aktivität des Nachfolgers zur korrekten Weiterleitung

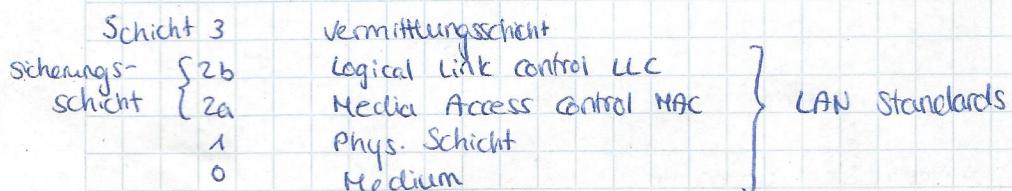
Zentrale Kontrolle kollisionsfreier Zugriff

Kontrolle durch zentralen Knoten (Polling)

\hookrightarrow jegliche Kommunikation über zentralen Knoten, keine direkte Komm.

Ethernet

Standardisierung



2b Logical Link Control LLC einheitlich für alle Medien

- Strukturierung der Übertragung
- Sicherung vor Übertragungsfehlern, Datenverlust
- Sicherung der Reihenfolge
- Flusskontrolle

2a Media Access Control MAC Zugangskontrolle für geteiltes Medium

- zufälliger Zugriff
- kollisionsfreier, kontrollierter Zugriff

Ethernet IEEE 802.3 Standard umfasst Schicht 1+2a

Medienzuteilung

- zeitmultiplex, variabel, zufälliger Zugriff
- CSMA/CD: Kollisionserkennung durch Mithören, exponentieller Backoff, 1-persistent

Netztopologie: Stern topologie
drahtgebunden, untersch. Medien

MAC-Adressen zB AA-BB-CC-00-44-77 durch Hersteller vergeben

u 8 Bit Adr. der Sicherungsschicht

↳ müssen im lokalen Netz eindeutig sein, flache Addressierung → neues Gerät bekommt neue nächste verfügbare Adr.

↳ lokale Nutzung zur Übertragung von Rahmen zw. benachbarten Interfaces

Adressarten

Unicast

- adr. spezielles Gerät
- ↳ nur dieses Gerät empfängt Daten

Multicast

- adr. Gruppe von Geräten
- alle Geräte dieser Gruppe empfangen Daten

Broadcast

- alle Geräte im Netz empfangen Daten

Anycast

- adr. Gruppe von Geräten → ausgewähltes Gerät daraus empfängt Daten

Protokoll

Ethernet Rahmen

min 64 Byte

PR (7)	SD (1)	DA (6)	SA (6)	Type (2)	Data (≤ 1500)	Pad (optional)	FCS (4)	Byte
Frame zur synchr. zur synchr. Beginn Übertragung		Zieladr.		Type = darüberliegendes Protokoll Length = Länge Nutzdaten		Padding	Frame Check Sequence	

Type/Length Feld

Ursprünglich: Type → Protokoll oberhalb von Ethernet

IEEE: Length → Länge Nutzdaten

real: beides → 0...1500 Länge 1516...65535 Typ

Exponentieller Backoff regelt Wartezeit bis zur nächsten Sendewahl.

Station wählt randomisiert Anzahl zu wartender Zeitschlitze aus

1. Kollision: 0/1 Zeitschlitze

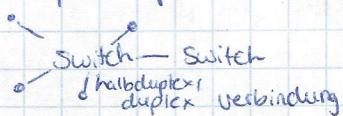
2. : 0/1/1/2/3 i max 16, danach Systemfehler angenommen

i. Kollision: 0/1...1 $2^i - 1$

Zeitschlitze: Dauer = min. Länge eines Rahmens

Switches: Stern topologie mit zentraler Sternkomponente

- Anschluss der Endsysteme per Punkt-zu-Punkt Medium



halfduplex: Nutzung von CSMA/CD

duplex: kein CSMA/CD erforderlich, separater Kanal für Send- und Empfangsrichtung

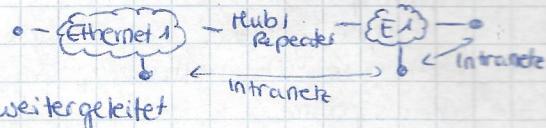
→ Ethernet-Protokoll auf den Links zum Switch

Switching: Weiterleitung von Eingangs- zu Ausgangsinterface, simultane Übertragung möglich

Kopplung von Ethernet

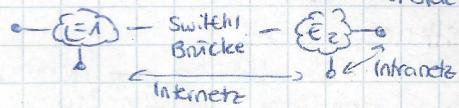
auf Schicht 1

- Zwischensysteme verstärken Signal
- Rahmen werden in alle Ethernet-Netzbereiche weitergeleitet



auf Schicht 2

- gekoppelt durch Switches und Brücken → transparent für Endsysteme: wissen nicht dass Switches/Brücken unterwegs
- Ethernet Rahmen weitergeleitet + ggf gepuffert
- Trennung Internet- und Intranet-Verkehr



Vernetzung von Switches

Plug-and-play: Switches selbstlernend

↳ etablierung Netztopologie ohne Schleifen: Spanning Tree Algo (minimaler Spannbaum)

↳ zielorientierte Weiterleitung zw. Endsystemen

Selbstlernende Switches Wege finden

(1) Switch empfängt Rahmen, kennt Zieladr. Nicht

↳ flutet Rahmen auf allen aktiven Interfaces

↳ lernt Lokation des Endsystems mit Ziel-Adr. → merkt sich Interface in Tabelle

(2) Switch empfängt Rahmen, kennt Zieladr. → Rahmen über entspr. Interface weiterleiten

Kollisionsdomäne Bereich des Netzes, der von Auftreten einer Kollision betroffen

Bus: gesamter Bus ist Kollisionsdomäne

Repeater: komplettes Netz

Brücke: getrennte Kollisionsdomänen links + rechts von Brücke

Steintopologie: > Hub: alles

> Switch: einzelne Geräte/Teilnetze

VLAN virtuelles Netz innerhalb physischem Netz → logische Trennung des Sicherheit gezielte Gruppierung durch virtuelle Leitungen Datenverkehrs auf Ethernet-Ebene
Flexibilität neue Geräte: keine Änderung am phys. Medium notwendig, nur Reorganisierung der logischen Medien

Performance Broadcastlast sinkt

Interface-basiert: Einzeller Switch arbeitet wie mehrere virtuelle Switches

↳ Verkehrsisolation: Rahmen können nur innerhalb ihrer virtuellen Switches agieren

↳ dynamische Zuweisung von Interfaces zu anderen VLANs

↳ Weiterleitung zw. VLANs: Routing über Router, der im Switch integriert

VLAN-Trunks Transportieren Rahmen zw. VLANs die durch mehrere phys. Switches erzeugt

↳ Rahmen keine normalen Ethernet-Rahmen

Pr	SP	DA	SA	Type	ID	Type	Data	Padl	FCS	Byte
(7)	(1)	(6)	(6)	(2)	(2)	(2)	(≤1500)	(opt.)	(4)	
ID von VLAN-Protokoll										

Vermittlungsschicht

Paketvermittlung

- Daten in Pakete gliedern, zu Zielsystemen weiterleiten, Pakete haben Metainformationen
- ↪ Abschnittsweise Übermittlung, Zwischenspeicherung ggf. in Warteschlangen der Zwischensysteme
- ↪ Puffnung falls Ausgangsinterface nicht frei
- ! Verlust von Paketen möglich durch begrenzte Pufferkapazitäten

Prinzip Vermittlungsschicht

Datenaustausch zw. Geräten, die nicht direkt benachbart sind

Pakete = Datagramme

Zwischensysteme = Router

→ Protokolle in allen Endsystemen und Routern

2 Grundlegende Aufgaben:

- > Wegewahl (Kontrollebene)
 - ↪ ermittelt Weg, den Pakete zurücklegen
- > Weiterleitung (Datenebene)
 - ↪ Pakete vom Reuteingang auf geeigneten Ausgang weiterleiten

Kontrollebene

- > Wegewahl = Routing: ermittelt Weg, den Pakete von Quell- zu Zielsystem zurücklegen
- > Hop = Übertragungsabschnitte: Setzt Pfad zusammen, ein Hop $\hat{=}$ Endsys - Router oder Router - Router
- > Routingprotokoll = Protokoll, das verantwortlich für Wegewahl
 - ↪ oberhalb Schicht 3 angesiedelt
- > Routingalgo = Algo für Wegewahl, umgesetzt von Routingprotokollen
- > Weiterleitungstabelle = Schnittstelle zw. Kontroll- und Datenebene, enthält Einträge für mögliche Ziele mit Interfaces, auf denen Datagramme weitergeleitet werden
- > Router = Zwischensystem, leitet Datagramme weiter, tauscht Infos mit anderen Routern:
Erstellung Routing- und Weiterleitungstabellen \rightarrow logisch zentrales Routing (SDN)

Rechnernetz als Graph

Knoten = Router

Übertragungsabschnitte = Kanten mit Kosten: Verzögerung, aktuelle Last, ...

Pfad = Weg = Folge von Knoten im Graph, wobei jeder Knoten max. einmal vorkommt

Routingverfahren

- nicht adaptiv: Routen ändern sich selten, seltener als Verkehrsänderungen
- adaptiv: Routenänderung in Abhängigkeit vom Verkehr / Netztopologie
 - ↪ berücksichtigt aktuellen Zustand des Netzes
 - ↪ Problem: Schleifen, Oszillationen, hohe Last durch Austausch Routing-Infos
 - ↪ periodisch oder direkte Reaktion auf Änderungen
- zentral: Wegewahl durch zentrale Komponente \rightarrow erstellt Weiterleitungstabellen in Zwischensystemen \rightarrow zentrale Komponente benötigt globale Sicht auf Netz
- dezentral: Routingentscheidung / Wegewahl dezentral in Zwischensystemen
 - ↪ keine zentrale Komponente nötig
 - > isoliert: kein Austausch mit anderen Routern, Entscheidung nur mit eigener Info
 - > verkilt: Austausch von Routinginfo mit Nachbarn

Statisches Routing isoliert
- zwischen sys. hat statische Weiterleitungs tabelle

Zentralisiertes Routing adaptiv

- zentrales Routing control Center RCC → bekommt periodisch Zustandsinfo aller Systeme
- berechnet damit optimale Wege, verteilt neue Weiterleitungsinfo an Router
 - ↪ Router trifft anhand dessen Routing-Entscheidung
- ⊕ RCC kann (theoretisch) perfekte Entscheidung treffen
- Systeme können aufwändige Routingberechnungen sparen
- ⊖ lange Berechnung für große Netze
- RCC Ausfall läuft ganzes Netz
- starke Belastung des RCC, sowie links nahe RCC

Flooding isoliert, nicht adaptiv

- jedes eingehende Datagramm auf jedem Interface weiterleiten (außer Empfangsinterface)
- Eindämmung: Duplicaterkennung (sequenznr.)
 - Zählen der Hops → Lebensdauer kontrollieren

Hop-Count isoliert, adaptiv

- jedes System versucht Datagramme so schnell wie möglich weiterzuleiten
 - ↪ Interface mit kürzester Queue, aber nicht auf Empfangsinterface

Verteiltes adaptives Routing Systeme tauschen Routinginfo mit Nachbarn

- Routing-Tabelle pro System: Infos über erreichbare Systeme, zB bevorzugter Übertragungsabschnitt zum Ziel, Schätzung Zeit/Entfernung zum Ziel
- Varianten: periodischer Austausch Routinginfo
- Austausch bei signifikanten Änderungen

Distanz-Vektor-Routing Metrik: Distanz

- Jeder Router kennt Distanz zu allen anderen Systemen im Netz
 - Problem: kürzerkostamater Weg bevorzugt gegenüber längrem schnellerem Weg
 - verteilt: jeder Router erhält Info von direkten Nachbarn, berechnet, sendet neue Infos an direkte Nachbarn
 - iterativ: verteilen, Berechnen von Info so lange bis keine neuen Infos mehr ausgetauscht
- Distanz-Vektor-Tabelle: in jedem System vorhanden, Zeile pro mögliches Ziel, Spalte für direkte Nachbarn

ALGORITHMUS: $O(n^2 k)$ $n = \text{knoten}$ $k = \text{kanten}$

Initialisierung: - ∀ Nachbarn von v : $D^x(x, v) = \infty$, $D^x(v, v) = c(x, v)$

- ∀ Ziele y : sende $\min_w D^w(y, w)$ zu jedem Nachbarn, wobei w alle Nachbarn enthält

$$D^x(a, b)$$

$$= x \rightarrow b \rightarrow a$$

Schleife: - Geänderte Übertragungsabschnitt Kosten: $c(x, y)$ ändert sich um d

$$\forall \text{ziele } y: D^x(y, v) = D^x(y, v) + d$$

- Update-Nachricht von Nachbar:

• kürzester Pfad von v zu Ziel y geändert zu "n"

$$D^x(y, v) = c(x, v) + "n" \text{ für dieses Ziel}$$

- neues $\min_w D^w(y, w)$ für Ziel y : sende diesen Wert an Nachbarn

→ Schlechte Neuigkeiten breiten sich langsam aus → u.U. Routing-Schleifen

→ Count-to-infinity-Problem

Lösung: Poised Reverse: Routing-Info wird Knoten trennen, wenn Weg über diesen Knoten kürzer

Link-State-Routing untersch. Routing-Metriken

- Berücksichtigt aktuelle Zustände der Netzanschlüsse
- jeder Router kennt komplett Netztopologie

Vorgehensweise:

- Systeme kennen anfangs nur direkte Nachbarn
- entdecken neue Nachbarn mittels spezieller Routingnachrichten
- Bestimmung Kosten zu direkten Nachbarn
- LINK STATE BROADCAST:

Fluten der Identität + Kosten zu direkten Nachbarn

↳ Systeme lernen Topologie \rightarrow alle Systeme haben identisches Wissen über Netz

\rightarrow nach Fluten + Berechnung kürzester Pfade in jedem System: schleifenfrei, in stabilem Zustand konvergiert

Dijkstra Algorithmus $O(n^2) \sim O(n \log n + kn)$

- $c(i,j)$ Kosten von i nach j , initial ∞
- $D(v)$ Kosten der Route von Quelle zur aktuellen Senke v
- $p(v)$ Vorgänger von v auf aktuell kürzestem Pfad zu v
- N Menge der Systeme, deren kürzester Pfad von der Quelle bekannt

Initialisierung: $N = \{\text{Quelle } A\}$, $D(v) = c(A, v) \quad \forall$ direkten Nachbarn von A

$$D(v) = \infty \quad \text{sonst}$$

Schleife: Finde System w mit $w \notin N$ und $D(w)$ ist Minimum

Füge w zu N hinzu

Erneuere $D(v) \quad \forall v \notin N, v$ direkter Nachbar von w

$$D(v) = \min(D(v), D(w) + c(w, v))$$

Problem: Oszillation

ZB weil Linkskosten je nach Zeit aktualisiert werden

Link-State vs. Distanz-Vektor

Komplexität Routingnachrichten:

Link-State: $O(n^2)$ n Systeme, \in Links, jedes System kennt Kosten aller Links

Distanz-Vektor: Änderungen nur an Nachbarn \rightarrow weniger Pakete

Konvergenzgeschw.

Link-State: $O(n^2)$ Komplexität, $O(nE)$ Pakete

\rightarrow schnelle Konvergenz, danach schleifenfrei, Oszillationen möglich

Distanz-Vektor: Konvergenz langsam, evtl. Routing-Schleifen (Count-to-infinity)

Robustheit

Link-State: Routenberechnung separat \rightarrow robuster

Distanz-Vektor: inkorrekte Pfade breiten sich aus

\Rightarrow L-S konvergiert schneller, robuster

D-V leichter implementierbar

Datenebene Weiterleitung der Daten vom Eingangs- zum Ausgangsinterface

IP-Adressen Eindeutige Identifikation aller angeschlossenen Geräte bzw. deren Interfaces

IPv4: 32 Bit → je 8 Bit Blöcke einzeln codiert, darstellen der Form 223.1.1.1

IPv6: 128 Bit

Gliederung in Subnetz-Teil, Endsystem-Teil

Format: a.b.c.d/x x = Anzahl Bits im Subnetz-Teil (erster Teil der IP-Adr.)

Subnetz IP-Adr. mit gleichem Subnetz-Teil → Interfaces können sich ohne Router erreichen

Internet Protokol IP verbindungsloser, unzuverlässiger Vermittlungsdienst
Weiterleitung IP-Datagramme

Fragmentierung, Reassemblierung Reassemblierung nur im Endsystem

Anpassung an versch. lange max. Länge der Rahmen unterliegender Netze (Maximum Transfer Unit MTU)

Flag-Felder in IP-Kopf: Bit 1 0/1 → darf nicht/darf fragmentiert werden

Bit 2 0 letztes Fragment / 1 weitere Fragmente

Fragment-Offset: Stelle an der Fragment in Datagramm einsetzen

Weiterleitung

→ gleiches lokales Netz/direkt verbunden: direkt an Empfänger schicken

→ ansonsten: Datagramm an Default-Router

→ Weiterleitungs-Tabelle: Angabe Zieladdr., Next-Hop-Router, Flags

Empfang IP-Datagramm

Überprüfe: u.a. korrekte Kopflänge, Datagrammlänge, Prüfsumme, Lebenszeit, ...

Fehler erkann.: Internet Control Message Protocol (ICMP) benachrichtigen

Eieruhr-Modell

IP einziges Protokoll auf Vermittlungsschicht → Interoperabilität höher, weniger untersch. Interfaces

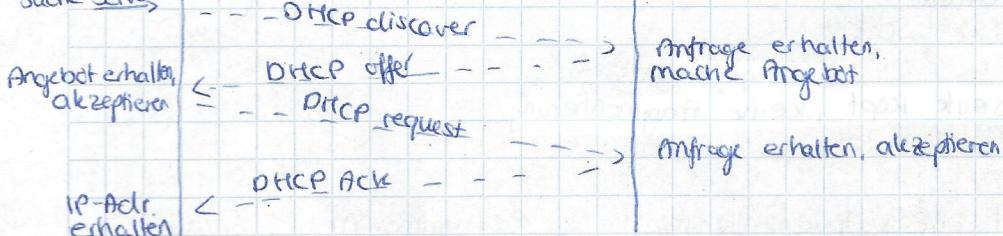
→ einfaches Protokoll: nutzbar für viele Netze

Addresszuweisung Manuell oder Dynamisch

→ Dynamic Host Configuration Protocol (DHCP):

DHCP Server liefert bei Anfrage IP-Adr.
neues Endsys

suche Server



→ Provider erhält Adr. Block von ICAAN Organisation zugewiesen

↳ Organisation bekommt Subnetz-Teil von seinem Provider

Internet Control Message Protocol (ICMP) nur schwerwiegende Fehler werden gemeldet
unterstützt Austausch von Fehlernachrichten, Statusanfragen, Zustandsinfo
Statusanfragen

→ Echo + Echoantwort: zur Überprüfung der Gerätfeaktivität, Echo-Antwort sendet
erhaltene Daten zurück

→ Zeitstempel + -antwort: Bestimmung Umlaufzeit (RTT), Bearbeitungszeit, Netzverzögerung

Format Übertragung im Datenfeld eines IP-Datagramms, Kennzeichnung durch "1" im Protokoll-Feld des IP-Kopfs über Datagramm

IP-Kopf	Type	Code	Checksum	Info
ICMP-Datagramm				

Type = Datagrammtyp,
Code = Genaue Beschreibung des Octagr.
Info = abh. von Type

ARP Address Resolution Protocol ermittelt MAC Adr. zu bekannter IP-Adr.

zielsystem im gleichen Subnetz, falls

$$\text{Ziel-IP-Adr.} \text{ AND } \text{Subnetzmuske} = \text{Eigene-IP-Adr. AND Subnetzmuske}$$

→ Dynamisches Lernen von Adr. Zuordnungen, Speichern in ARP-Cache: kleine Tabelle pro System <IP-Adr., MAC-Adr., Max. Lebenszeit> Einträge
↳ kein Netzadmind erforderlich

Adr. Auflösung lokal (gleiches Subnetz) A sendet an B

- Eintrag in Cache: Paket weiterleiten, Timeout refresh

- nicht in Cache: Broadcast ARP-Request (inkl. Adr. von B)

→ jeder Knoten liest Request, prüft IP-Adr.

→ eigene 2 ARP-Reply → MAC Adr. von B in Tabelle eintragen

Anderes Subnetz

- A sendet ARP-Request an Router R. A sendet Datagramm an IP-Adr. von B und MAC-Adr. von R, R empfängt, setzt Ziel-, Sender-MAC Adr. auf Adr. von B, R → leitet Datagramm in anderem Subnetz weiter

Security

Angriffe: DOS, Man-in-the-Middle (gibt gefälschte ARP-Replies)

Übersicht Vermittlungsschicht in Endsystem, Router vorhanden

Routing-Protokolle

Transportsschicht: UDP, TCP

Vermittlungsschicht

Protokoll IP

- Adressierung
- Datagramme
- Paketverarbeitung

weiterleitungs-tabelle

Protokoll ICMP

- Fehlernachrichten
- Signalisierung zw. Routern

Adr. Auflösung
- ARP, RARP

Sicherungsschicht

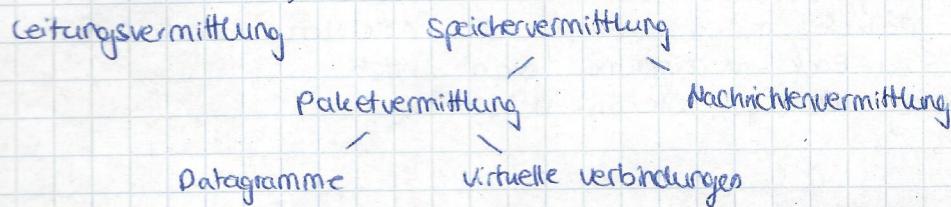
physikalische Schicht

IPv6 128 Bit Adr.

Adr. Raum von IPv4 zu klein

Datagramm: 40 Byte Kopf, keine Fragmentierung, keine Prüfsumme mehr, Optionen als Erweiterungsköpfe

Vermittlungstechniken



Paketvermittlung Daten in Pakete gegliedert, zu Zielsystem in diesen Paketen
 ↳ besitzen Metainformationen (Kontrollinfo)

- Adr. info: Zieladr. $\xrightarrow{\text{z.B. Paket}}$ lokale Kennung bei virtuellen Verbindungen
- Reihenfolgetauschungen möglich \rightarrow versch. Wege für Pakete

- Abschnittsweise Übermittlung
- Zwischen Speicherung in Warteschlangen in Zwischen systemen
 ↳ Verlust von Paketen möglich: Puffer voll
- Zeitmultiplex, keine Ressourcenreservierung
- statistisches Multiplexing \rightarrow Senden bei Bedarf, keine feste Zeitschlitz-Zuordnung

Datagrammvermittlung

- Datagramm als isolierte Einheit
- Zieladr. in jedem Datagramm \rightarrow verbindungslos (kein Aufbau / Abbau nötig)
- Datagramme i. d. R. auf untersch. Wegen \rightarrow Reihenfolgevertauschung
- Weiterleitungstabelle in Zwischen systemen

Virtuelle Verbindungen

- "Virtuelle Leitung" = fester Pfad zw. Endsystemen
 ↳ alle Pakete gleicher Pfad \rightarrow richtige Reihenfolge
- Vermittlung anhand der Paket-Kennung
 - verbindungsorientiert: Zieladr. nur beim Aufbau nötig
- Verbindungsaufbau \rightarrow Datenübertragung \rightarrow Verbindungsabbau

virtuelle Verb. durch

Festlegung von Kennungen
 in zw. Sys.

Vergleich

	VV	Datagramme
Zieladr.	während Aufbau nötig	jedes Datagramm \rightarrow Overhead
Reihenfolge	✓	✗
Verbindung	Auf- / Abbau \rightarrow zeitl. + Informationsoverhead	verbindungslos
Netzkomplexität, -funktionalität	Quality-of-Service einfacher mehr Funktionalität im Netz	Keine Zustandshaltung mehr Funktl. im Endsys
Bsp.	HPLS	Internet

- Nachrichtenvermittlung: Nachrichten der Anwendung Hop-by-Hop zum Zielsystem
- Segmentierung + Reassemblierung in Paketen
 - ↳ alle Pakete müssen an gleicher nächsten Zwischensys.
 - Ende-zu-Ende-Verzögerung recht hoch
 - Anwendung: Delay-Tolerant Networks

Leistungsvermittlung

- durchgehender Übertragungskanal für Verbindung mit konstanter Bandbreite verfügbar
- ↳ einzelne Übertragungsabschnitte in zw. Systemen miteinander verknüpft
 - Zustandshaltung in zw. System
 - Wegewahl bei Verbindungsauflaufbau
 - statisches Multiplexen möglich: Frequenz- / Zeitmultiplex
 - Reihenfolgentreu
 - keine Schwankungen durch Puffer
 - Bsp.: ISDN, ATM Telekommunikationsnetze (digital)

Software-Defined Networking SON

Eigenschaften

- (1) Separierung Kontroll- und Datenebene
- (2) Flow-basierte Paketweiterleitung
- (3) Logik ausgelagert an externen Controller
- (4) Programmierbarkeit des Netzes

aktueller "Standard": OpenFlow

↓
Schnittstelle zw. Switch
Controller
→ Programmieren von Flows

Traditionelles IP-Routing: Kontroll- & Datenebene in jedem Router

- ① Ausfallsicherheit
- ② Proprietär: herstellerspez. Schnittstellen
- schnelle Reaktion
- unflexibel: neue Funktionen schwierig
- bewährtes Konzept
- teuer

SON → Kontrollebene in zentralem Controller für alle Router

↳ offenes Interface

- ③ Controller hat Übersicht über gesamtes Netz
- neue Funktionalität als Software in Controller
- Trennung Kontroll-, Datenebene

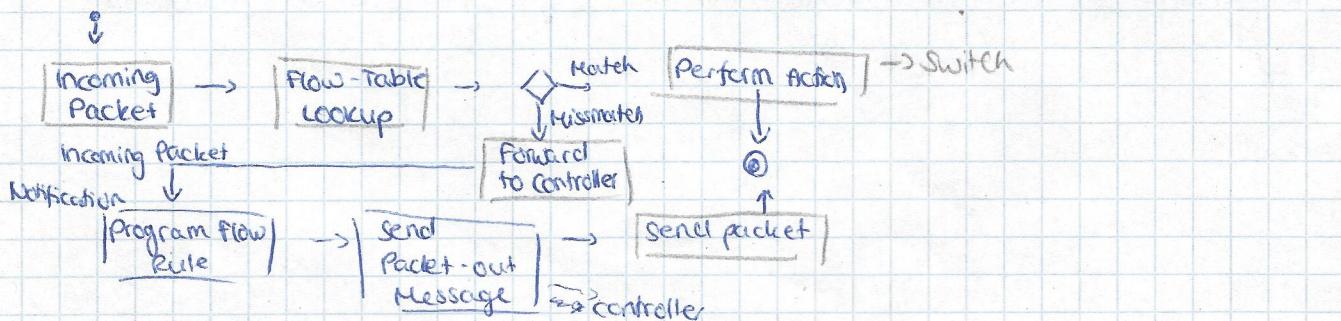
- ④ Skalierbarkeit

Single point of failure

Kommunikation mit Controller langsam

Flows

- Traditioneller IP-Router: keine Flows, Weiterleitung über Ziel-IP-Adr.
- SDN-fähiger Switch: Weiterleitung über Flortabelle, kein Eintrag: Controller programmiert entsprechenden Flow auf Switch



Transportschicht

Protokolle in Endsystemen

Datenaustausch zw. Anwendungsprozessen \rightarrow Nutzer-zu-Nutzer-Kommunikation
Packet = Segmente / Datagramme

- zuverlässige, unzuverlässige Dienste
- Pufferung der Daten in Endsystemen
- Adressierung von Anwendungsprozessen
- Daten der Transportschicht in Datagrammen der Vermittlungsschicht übertragen

Nutzer-zu-Nutzer = Anwendung zu Anwendung
Ende-zu-Ende = Gerät zu Gerät

Client-Server-Architektur

Server = System, das ständig betriebsbereit, erreichbar ist; Clients Anwendungsdienste bereitstellt
 \rightarrow zentralisierte Architektur, idR statische IP-Adr.

Client = nimmt Anwendungsdienste eines Servers in Anspruch, nicht immer in Betrieb / erreichbar \rightarrow dynamische IP-Adr.
 \rightarrow kommuniziert nur mit Server, nicht mit anderen Clients

Adressierung: Ports : Nummer der Länge 16 Bit, lokal (pro Endsys.) eindeutig
Anfragebsp.: webserver über HTTP (Port 80)
CIP-Adrs: < Ports > \sim 129.13.61.63 : 80
 \rightarrow IP-Adr. bestimmt Interface des Endsys.
Port bestimmt Anwendungsprozess in Endsys.

Portnummernvergabe:

0-1023 well known port numbers

- Vergabe durch IANA

- Nutzung durch system- / privilegierte Prozesse

1024 - 49151 registered port numbers

- Registrierung durch IANA

- Nutzung durch Nutzer- / gewöhnliche Prozesse

49152 - 65535 dynamic and/or private ports

- keine Registrierung vorgesehen

Datenstrom: Daten, die zw. Anwendungsprozessen ausgetauscht

Eindeutige Identifikation durch

(ziel-IP-Adr., Ziel-Port, Quell-IP-Adr., Quell-Port, Transportprotokoll)

UDP

- Multiplexen / Demultiplexen von Segmenten für Prozesse
- Prüfsumme zur Bitfehlerprüfung
- keine Verbindung \rightarrow kein Auf-/Abbau
- unzuverlässiger Dienst
- UDP Segment: geringer Overhead durch kleinen Transportkopf
- unregulierte Senken \rightarrow best effort: keine Zusagen über Auslieferung der Daten

Quell-Port	Ziel-Port
Länge	Prüfsumme
Daten	

UDP Segment

Einsatz: interaktive Multimedia-Anwendungen, z.B. VoIP / oder DNS

Prüfsumme: Σ alle übertragenen Werte (interpretieren als 16 Bit Integer) \rightarrow keine Reihenfolgenverluste erkennbar
Addition mit 1-komplement

UOP: Prüfsumme über Segment inkl. Kopf + Pseudohandler
 → mit IPv4 optional, IPv6 Pflicht

Pseudohandler: Quell-/Ziel-IP-Adr., Protokoll, Segmentlänge
 → Fehler in IP-Adr. erkennbar

TCP

- zuverlässiger Dienst, verbindungsorientiert
- erhält von Anwendung Bytestrom, übergibt TCP-Segmente an IP

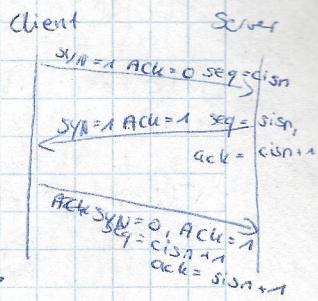
Bytestrom → TCP-Segment

"TCP should send at its own convenience"
 oder:

MSS: Maximum Segment Size, zB Größe, sodass IP nicht fragmentiert

Push-Flag: Sender verkündigt sofortiges senden

Zeitgeber: Zeitintervall



Fehlerkontrolle

Sequenznummern: pro Byte statt pro Segment, initiale Sequenznr. zufällig gewählt vom Endsystem

Quittung: positive kumulative Quittungen, Sequenznr. des nächsten Bytes, das erwartet wird

Flusskontrolle

Empfänger reserviert Pufferplatz pro Verbindung: Rcv Buffer

↪ Rcv Window: noch freier Pufferplatz (Empfangsfenster)

↪ Feld im Kopf des TCP-Segments

Empfänger informiert Sender

→ Sender sendet nicht mehr unbestätigte Daten als in Rcv Window passen

Es muss gelten: > last Byte Rcvd - last Byte Read <= Rcv Buffer

→ Rcv Window = Rcv Buffer - (last Byte Rcvd - last Byte Read)

> last Byte Send - last Byte Acked <= Rcv Window

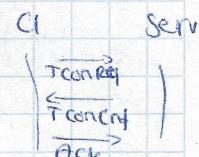
Verbindungsverwaltung

Aufbau: 3-wege-Handshake, Client initiiert, Server beantwortet

- Connection request / conf führen keine Nutzdaten mit sich

- Bekanntgabe Größe Rcv Window / Buffer

- Festlegung init. Seq. nr.



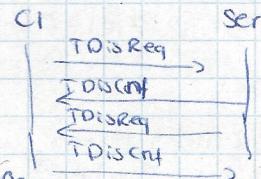
Abbau: von Client oder Server initiiert
 nach letztem ACK: Wörter beider lokaler Kontext gelöscht

> Ordnungsgemäß: 4-wege-Handshake

Richtungen werden unabh. geschlossen

> Verbindungsabbau: Reset, Verbindung unmittelbar schließen

↪ Kontext unmittelbar gelöscht



Staukontrolle: Stau = Pufferüberlauf → Paketverlust → Sendewahl. → mehr Stau

Staukontrollfenster Cwnd: Last Byte Send - last Byte Acked <= min { Cwnd, Rcv Window }

+ Schwellenwert (SSThresh)

Start: Cwnd = 1 MSS → Slow-Start: pro empfangene Quittung Cwnd += 1

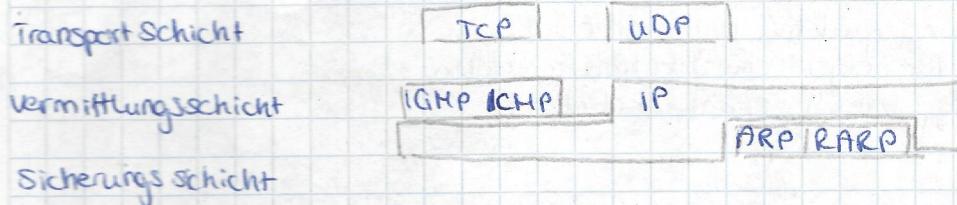
→ Cwnd > SSThresh + Quittung rechtzeitig: Congestion Avoidance Cwnd += 1/Cwnd

→ Quittung nicht rechtzeitig: Stau vermutet: SSThresh = max (FlightSize/2, 2 * MSS)

↪ Cwnd reset, Cwnd = 1MSS → neue Slow-Start-Phase

Oetten die gesendet, aber nicht quittiert

Zusammen Spiel Protokolle / Schichten



TCP = Transmission Control Protocol : zuverlässiger Transportdienst

UDP = User Datagram Protocol : unzuverlässiger Transportdienst

IP = Internet Protocol : wegwahl, unzuverlässige Datagrammübertragung

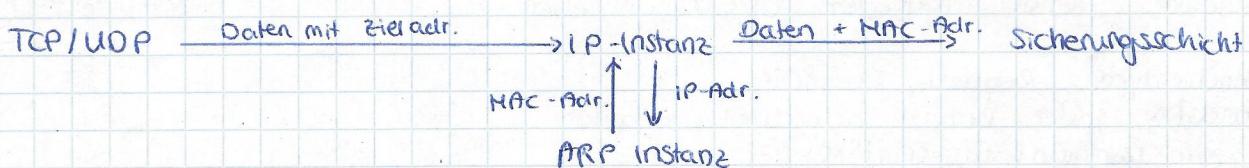
ICMP = Internet Control Message protocol : unterstützt Austausch von Kontrollinfo innerhalb Vermittlungsschicht

IGMP = Internet Group Management protocol : unterstützt Verwaltung von Kommunikationsgruppen

ARP = Address Resolution protocol : Unterstützt Zuordnung von IP-Adressen zu entspr. Adressen der Sicherungsschicht (MAC)

RARP = Reverse ARP : Umkehrfunktion von ARP

Senden



Empfangen

Unterscheidung zw. versch. Protokollen anhand Protokollfeld im IP-Kopf

↳ IP-Instanz reicht Daten an TCP / UDP Instanz weiter

Anwendungsschicht

Instanzen tauschen Nachrichten aus

Verteilte Anwendung = Anwendung, die auf mehreren Endsystemen ausgeführt wird
↳ Anwendungsprozesse nutzen Rechnernetze zum Austausch über versch. Endsysteme

IPC = Nachrichtenaustausch auf gleichem Endsystem zw. Prozessen

Client-Server-Anwendung = verteilte Anwendung wobei Prozesse entweder Client- oder Serverprozesse sind
↳ SPS ständig in Betrieb

CPS senden Requests an SPS, SPS senden Antworten an CPS
CPS kommunizieren nicht direkt miteinander

Peer-to-Peer-Anwendung = alle Anwendungsinstanzen sind sowohl Client als auch Server
↳ alle Peers gleichberechtigt, kein zentraler Server

Socket-Interface

Programmierschnittstelle für verteilte Anwendungen

- von OS bereitgestellt

- Anwendungsprozess sendet / empfängt Nachrichten zum / von Socket

- Sockets werden von Anwendungen erstellt, genutzt, freigegeben

↳ Adr. über Portnummern

↳ ermöglichen Lesen / Schreiben von Daten

↳ können Daten puffern

TCP-Sockets

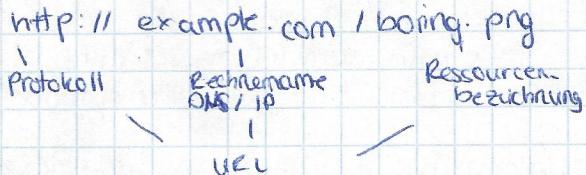
Server: 2 sockets

- Welcome-Socket: ständig aktiv, Anfragestelle
- Connection-Socket: für eigentlichen Datentransfer etabliert

Client: initiiert Verbindung, kreiert Client-Socket

Web und HTTP

Website = Basis - HTML-Dokument + Objekte → jedes Objekt über URL adressierbar



HTTP Hypertext Transfer Protocol

- Protokoll der Anwendungsschicht
- Client-Server-Architektur

Client: Browser, der Web-Objekte über HTTP anfordert, empfängt

Server: sendet angforderte Objekt an Client

- Nachrichtentypen: Request, Response
- Zustandslos: jeder Request individuell bearbeitet
- TCP zur Kommunikation (Port 80)
- versch. Methoden bei HTTP-Anfrage: get, post, put, delete, ...
- Statuscodes als Indikator für Verarbeitung → Erfolg / Fehlschlag + Gründe

Verbindungen

> non-persistent HTTP:

pro Objekt eine TCP-Verbindung (inkl. Auf-/Abbau)

→ Antwortzeit pro Objekt: $2 \cdot RTT + ts$ ($1 \cdot RTT$ für Aufbau, RTT für Anfrage, ts für Senden)

⊖ lange Antwortzeit, Betriebssystem-Overhead (pro Anfrage ein Socket!)

↪ parallele TCP-Verbindungen um alle Objekte zu laden

> persistent HTTP: TCP-Verbindung offen halten → mehrere Objekte senden

↪ Abbau nach Zeitintervall

⊕ nur eine RTT pro Objekt

Cookies eigentlich: HTTP-Server Zustandslos

Cookie = Textdatei mit Zustandsinfo, auf Client übertragen um Server zu entlasten

↪ Nutzer Einstellungen, -Authentifizierung

- vom Browser auf Endsys. verwaltet

- Server: Funktion um Cookies zu verstehen, zB DB mit Session-IDs

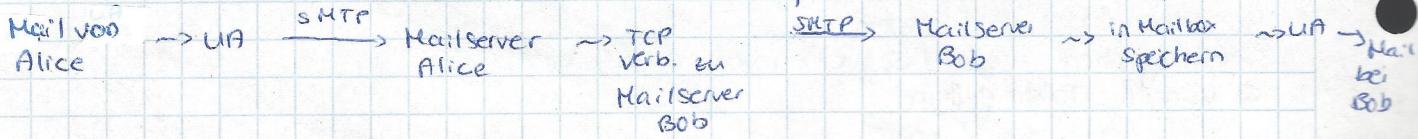
- wird in Response vom Server gesetzt, in nachfolgenden Requests genutzt

E-Mail

User Agent: lesen, senden, weiterleiten z.B. Outlook, Apple Mail

Mail server: Mail Transfer Agent, Mail Delivery Agent, Mailboxen der Nutzer

Simple Mail Transfer protocol: Client-Server-Architektur, Transfer Mail von UA zu Mailserver, Transfer zw. Mailservern



Simple Mail Transfer Protocol SMTP

- 3 Phasen: Handshake → Übermittlung Nachrichten → Abschluss
- Command Response ähnlich zu HTTP (HTTP: Pull-Protokoll, SMTP: Push-Protokoll)
- 7-bit-ASCII Nachrichten
- TCP: zuverlässige Übertragung zum Server (! nicht zum Nutzer)

Format

Header + Body

Problem: keine Übertragung von Binärdateien / Programmen
nur 7-bit-ASCII

MIME Multi-Purpose Internet Mail Extensions

erweitert Header um Formatinfos

Content-Type: definiert Typ des Mail-Inhalts

Content-Transfer-Encoding: definiert Transfersyntax, in der Daten des Hauptteils übertragen

Postfachabfrage

- POP3 (Post office Protocol 3): Protokoll zum Abholen der gespeicherten Nachrichten von Mailbox auf Mailserver
- ↳ verwaltet Nachrichten im UA ! PW im Klartext übertragen

- IMAP (Interactive Mail Access Protocol) verwaltet Nachrichten auf Mailserver

Sicherheit

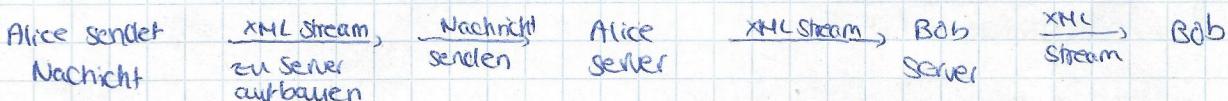
Fehlende Nutzer-zu-Nutzer-Sicherheit

↳ Angriffe: Mitlesen, Manipulation, fälschen von Nachrichten

WhatsApp zentraler Server, Protokoll ≈ XMPP

XMPP (Extensible Messaging and Presence Protocol)

- nahezu Echtzeit Datenaustausch
- Client-Server-Architektur: mehrere Server verbinden sich untereinander
- XML als Nachrichtenformat
- Adressformat: Nutzer durch Server + Username
Client durch Nutzer + bsp. Gerät



Domain Name System DNS

Protokoll der Anwendungsschicht

Zuordnung Name ↔ IP-Adr.

→ Basisdienst im Internet

- verteiltes System aus DNS-Servern

↳ kein Server kennt alle Zuordnungen

Autoritativer Server: liefert maßgebliche Antwort

Namensraum

hierarchisch: Subdomäne . Domäne . Top-Level-Domäne → conder-Domänen
Generische (durch ICANN festgelegt)

Anfrage

Endsystem kennt Namen nicht → lokalen Server fragen

↳ Antwort aus Datenbank (falls autoritativ) oder Cache

↳ Befragung anderer Server falls Antwort nicht kennt

Rekursive Anfragen

DNS-Server befragt weitere Server bis er Antwort lernt
 ↗ weniger Aufwand für Client

Iterative Anfragen

Antwort: Verweis auf anderen Server → Client befragt weitere Server bis Antwort
 ↗ weniger Aufwand für angefragten Server

Typisch

Client fragt lokalen Server rekursiv
 lokaler Server arbeitet iterativ

Hierarchie DNS - Server

Root - Server (weltweit nur wenige)

Tcp-level - Domain Server

!

Authoritative DNS - Server → jedes Endsys. bei einem registriert

!

Lokaler DNS - Server

- es gilt pro Domäne:
 - IP - Adr. mind. eines Root - Servers bekannt
 - Name - Server für nächst tiefere Ebene bekannt
 - enthält Einträge für die er autoritativ
 - Ergebnisse von Anfragen zwischengespeichert

Netz Sicherheit

Netzsicherheit = Maßnahmen zur Unterstützung der Sicherheit in Rechnernetzen
 technische + organisatorische Maßnahmen

Safety = System funktioniert "korrekt", Ist - Funktionalität = Soll - Funktionalität
 Security = System vor böswilligen Angriffen + deren Auswirkungen geschützt

Bedrohung = Ereignis / Reihe von Ereignissen, die zur Gefährdung einer/mehrerer Schutzziele führt

Angriff = gezielte Realisierung einer Bedrohung

Angreifermodell = Fähigkeit eines Angreifers einen Angriff auf ein System durchzuführen

Passive Angriffe: Abhören, Speichern von übertragenen Paketen

aktive Angriffe: Replay (Pakete duplizieren, neu senden), Ent�gen / löschen / Modifizieren von Paketen

Dolev - Yao - Angreifer: omnipräsent im Netz, kann sämtliche Kommunikation abhören, eigene Pakete erzeugen + versenden, fremde Pakete modifizieren
 kein Zugriff auf Endsys., kann nicht ver-/entschlüsseln ohne Schlüssel

Schutzziele CIA

Anforderungen an System/Komponente die erfüllt werden müssen um schützenswerte Güter vor Bedrohung zu schützen

Confidentiality Vertraulichkeit

Keine unautorisierte Informationsgewinnung
Umsetzung: Verschlüsselung

Integrity Integrität

- > starke Integrität: nicht möglich unautorisiert Daten zu manipulieren
- > schwache Integrität: unaut. Man. wird bemerkt \rightarrow im Internet umsetzbar
- Umsetzung: Tamper Proof Module (TPM)
Message Authentication Codes (MAC)

Authentizität

manipulationssichere Identifikation von Instanzen

- Echtheit von Instanzen \rightarrow Kommunikation wirklich mit XY
- Echtheit von Daten \rightarrow Daten wirklich von XY

Umsetzung: Zertifikate, Signaturen, gem. Geheimnis

Verschlüsselung

Kryptographie untersucht / entwickelt Verfahren zur Unkenntlichmachung von Klartexten zu Ciphertexten

Symmetrisch = beide Instanzen gemeinsamen Schlüssel pk für ver-, entschl.
asymmetrisch = geheimer + öffentl. Schlüssel

Symmetrische Verschlüsselung zB AES

Blockchiffren Daten als Blöcke der Länge pk , Blockweise verschlüsseln

Streamchiffren Bit-/Zeichenweise verschlüsseln, Schlüsselstrom aus seed

\rightarrow muss zufällig sein / aussehen

$$C_i = X_i \oplus S_i$$

$$X_i = C_i \oplus S_i$$

asymmetrische Verschlüsselung

2 Schlüssel: öffentlich: allen bekannt \rightarrow Verschlüsselung pk
privat: nur einem bekannt \rightarrow Entschlüsselung sk

$$C = \text{Enc}_{pk}(m)$$

$$m = \text{dec}_{sk}(C)$$

sk aus pk nicht einfach berechenbar \rightarrow Einwegfunktionen nötig

RSA Problem: Integer-Faktorisierung

P, Q groß, prim $P \neq Q$

$$N = P \cdot Q$$

$$\varphi(N) = (P-1)(Q-1)$$

$$e > 2, \text{ ggT}(e, \varphi(N)) = 1$$

$$d = e^{-1} \bmod \varphi(N)$$

$$pk = (N, e)$$

$$sk = (N, d)$$

langsam, Nutzung für Schlüsselaustausch
für symm. Verfahren

$$\text{Enc}(m) = m^e \bmod N = C$$

$$\text{Dec}(C) = C^d \bmod N = m$$

$$c, m \in \{0, 1, \dots, N-1\} = \mathbb{Z}_N$$

m als Integer interpretiert

\rightarrow im Zweifel aufteilen in Blöcke

$$\text{Blockgröße} \leq \log_2(N) + 1$$

Schlüssel austausch

gem. Schlüssel teilen für symm. Verfahren

Diffie-Hellman dlog Problem

Gemeinsames Erzeugerelement g einer zykl. Gruppe mit Ordnung p

A wählt $x \in [2, p-2]$, sendet $g^x \bmod p$ an B

B wählt $y \in [2, p-2]$, sendet $g^y \bmod p$ an A

→ gem. Schlüssel ist $(g^x)^y = (g^y)^x = g^{xy}$

- (+) keine Infrastrukturunterstützung nötig
- (-) anonymer Austausch → man-in-the-middle Angriff möglich

Integritäts sicherung

Hashfunktionen: $H: \{0,1\}^* \rightarrow \{0,1\}^k$

- > Einwegfunktion
- > Kollisionsresistenz (enthaltet target kollisionsresistent)
- > liefert Wert fester Länge
- > effizient

Message Authentication Code MAC

berechne $H(m)$ zu Nachricht m , wobei m

berechne Hash über Daten und gemeinsamen Schlüssel $H(m, k)$
zur Überprüfung: neu berechnen mit $H(m, k)$ und k

Digitale Signatur (asym. Verfahren)

Signiere Nachricht mit $sk \rightarrow \sigma = \text{Sig}(sk, m)$

Verifizierte mit $pk \rightarrow \text{Ver}(pk, m, \sigma) = 1 \Leftrightarrow \sigma$ gültig für m

zB mit RSA $\text{Sig}(sk, m) = m^d \bmod N$
 $\text{Ver}(pk, m, \sigma) = 1 \Leftrightarrow \sigma^e = m \bmod N$

Authentifizierung

symmetrisch:

Challenge - Response - Verfahren basierend auf gem. Geheimnis k

asymmetrisch:

Digitale Zertifikate → vertraue auf vertrauenswürdige Dritte → CA

Zertifikat = Instanz bestätigt Sachverhalt mittels Signatur eines dig. Dokuments

ID-Zertifikate: öffentl. Schlüssel → eindeutige Identität
enthalt: ↓ + Namenskennung der Zertifikatinstanz, Signatur der CA

E-Mail Sicherheit Ende-zu-Ende + Nutzer-zu-Nutzer

Mail verschlüsselung:

(1) $\text{Enc}(kr, m)$ (kr zufällig)

(2) $\text{Enc}(pk_B, kr)$

(3) sende beides an B

Entschlüsselung:

(1) $\text{Dec}(sk_B, \text{Enc}(pk_B, kr)) = kr$

(2) $m = \text{Dec}(kr, \text{Enc}(kr, m))$

→ benötigt öffentliche Schlüssel → Schlüsselaustausch + Authentizität

Lösungen: nutze Public key Infrastruktur

Infrastruktursicherheit

Firewall schützt vor Eindringen unerwünschter Pakete, isoliert Netzbereiche
realisieren Zugriffskontrolle
↳ Paket weiterleiten oder verwerten?

Zustandslose Paketfilterung

- Paketfilterung ohne Zustandsinfo
- basierend z.B. auf IP-Adr., Ports, Protokollen

Zustandsbehaftete Paketfilterung

- Filterung auf Basis von Zustandsinfos (Timeout für Zustandsinfos)
- Bsp. Prüfung ob TCP-Segmente zueinander passen

Access Control List

Liste von Regeln, von oben nach unten priorisiert
kann für zustandslose oder zust. beh. Filterung angewandt werden

Grenzen / Probleme

- Absender-Info kann gefälscht sein
- Verfügbarkeit: Single-Point-of-Failure, Filterung schützt nicht vor Überlastung
- nicht 100%ig sicher → Hardware Hintertüren

Intrusion Detection und Prevention erkennen + Verhindern von Angriffen

Intrusion Detection System IDS

können bekannte Angriffe erkennen → Deep Packet Inspection: Analyse der Nutzdaten
- regelbasierte Korrelation zw. Sitzungen
- Nutzung von Sensoren im Netz

Intrusion Prevention System IPS

- automatisierte Aktionen bei Angriff
- Nutzer vom Netz trennen
- kritische Systeme abschalten
- ...

können Angriffe verhindern

Anomaly Detection

- entdeckt unbekannte Angriffe
- kontinuierliches Monitoring
- Abweichung von Normalverhalten melden → machine learning

Organisatorische Maßnahmen

Netzsicherheit nicht allein durch Technik erreichbar
→ Faktor Mensch

Sicherheit ist kein Zustand, sondern ein Prozess!