Sicherungsschicht
(Fortsetzung)

Automatic Repeat Request (ARQ)

Bei Übertragungsfehlern kann der Empfänger den Sender informieren:

- NACK (Negative ACK)
- Kein ACK, dadurch Zeitüberschreitung beim Sender, der damit erneut versucht.
- Zeitüberschreitung muß in jedem Fall überwacht werden, da ein ACK/NACK verloren gehen kann.

Sender kann jeden nicht quittierten Rahmen erneut übertragen, behält dafür eine Kopie.

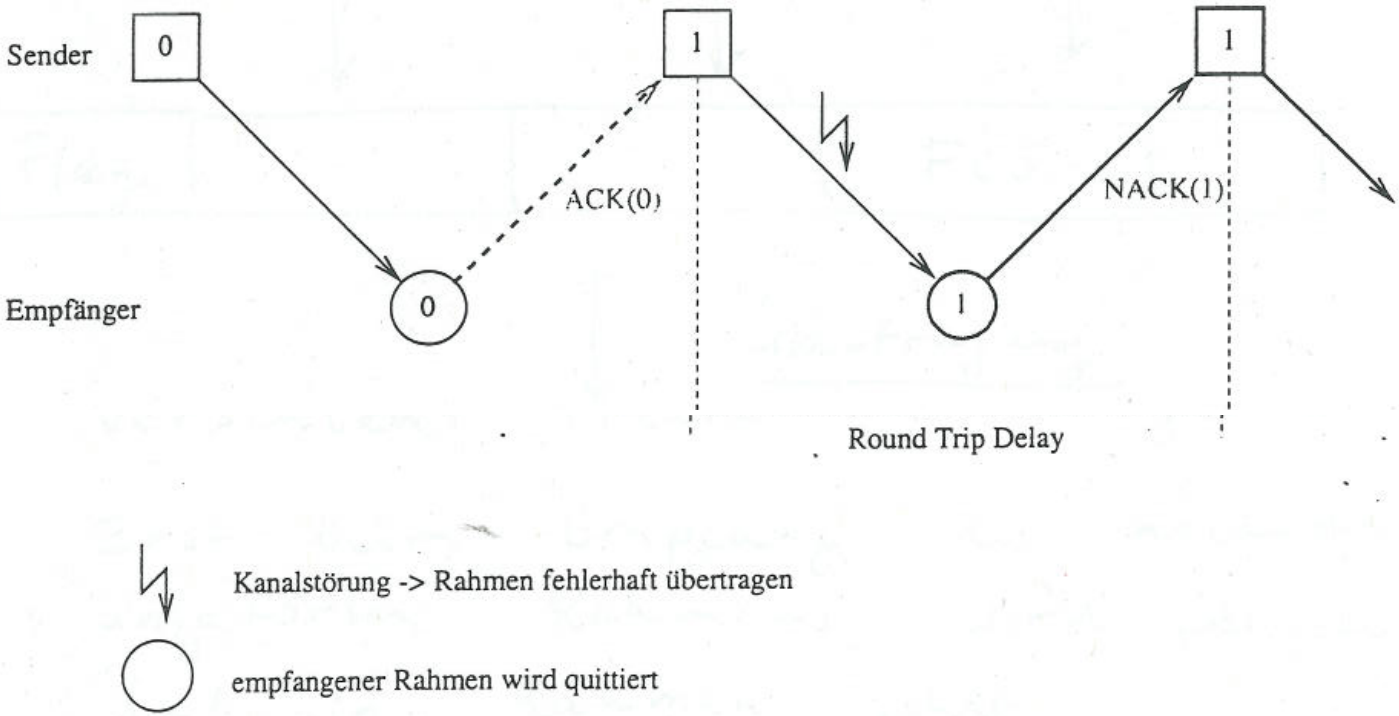
- Sender empfängt ACK(k), kann Rahmen bis Nummer $k - 1$ freigeben.
- Andernfalls erneute Übertragung einiger Rahmen (abhängig vom Protokoll)

ARQ Typen

- Stop-and-Wait ARQ
- Go-Back-*n* ARQ
- Selective Repeat ARQ

Stop-and-Wait ARQ

- Empfänger besitzt keinen Puffer.
- Es werden nur ein neuer Rahmen geschickt, wenn für den letzten ein ACK empfangen wurde



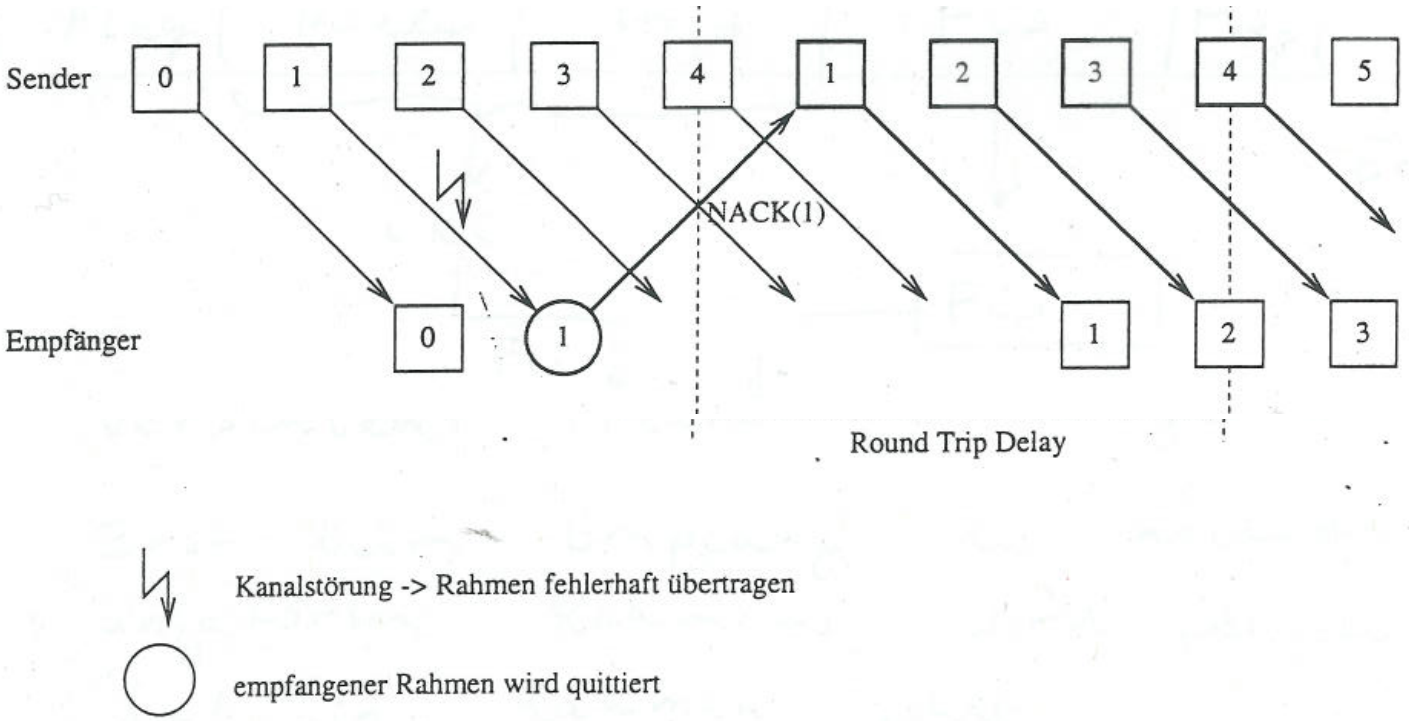
Stop-and-Wait ARQ

Wir können drei Fehlerfälle unterscheiden:

- Rahmen wird fehlerhaft übertragen.
  NACK löst erneute Übertragung aus.
- ACK/NACK geht verloren.
  Zeitüberschreitung löst erneute Übertragung aus.
- Datenrahmen geht verloren.
  Zeitüberschreitung löst erneute Übertragung aus.

Go-Back-*n* ARQ

- Empfänger hat Sliding Window mit $n - 1$ Plätzen.
- Es werden nur Rahmen in der richtigen Reihenfolge akzeptiert.
- Out-of-Order Rahmen werden verworfen.



Go-Back-n ARQ

Der Empfänger hat ein Sliding Window mit $n-1$ Plätzen, akzeptiert aber nur fehlerfreie Rahmen in der richtigen Reihenfolge.

Die folgenden Fälle sind relevant:

- Fehlerhaft übertragener Datenrahmen k

  Der Empfänger sendet NACK(k) und verwirft alle noch folgenden Rahmen, bis Rahmen k korrekt empfangen ist.

  Der Sender gibt alle Rahmen bis $k-1$ frei und sendet Rahmen k und folgende erneut.

Go-Back-n ARQ

Weitere Fehlerfälle:

- Verlorener Datenrahmen

  Der Empfänger sieht Rahmen $k+1$ ohne Rahmen k zu haben, sendet daher NACK(k) und verwirft die Rahmen $k+1, \ldots$

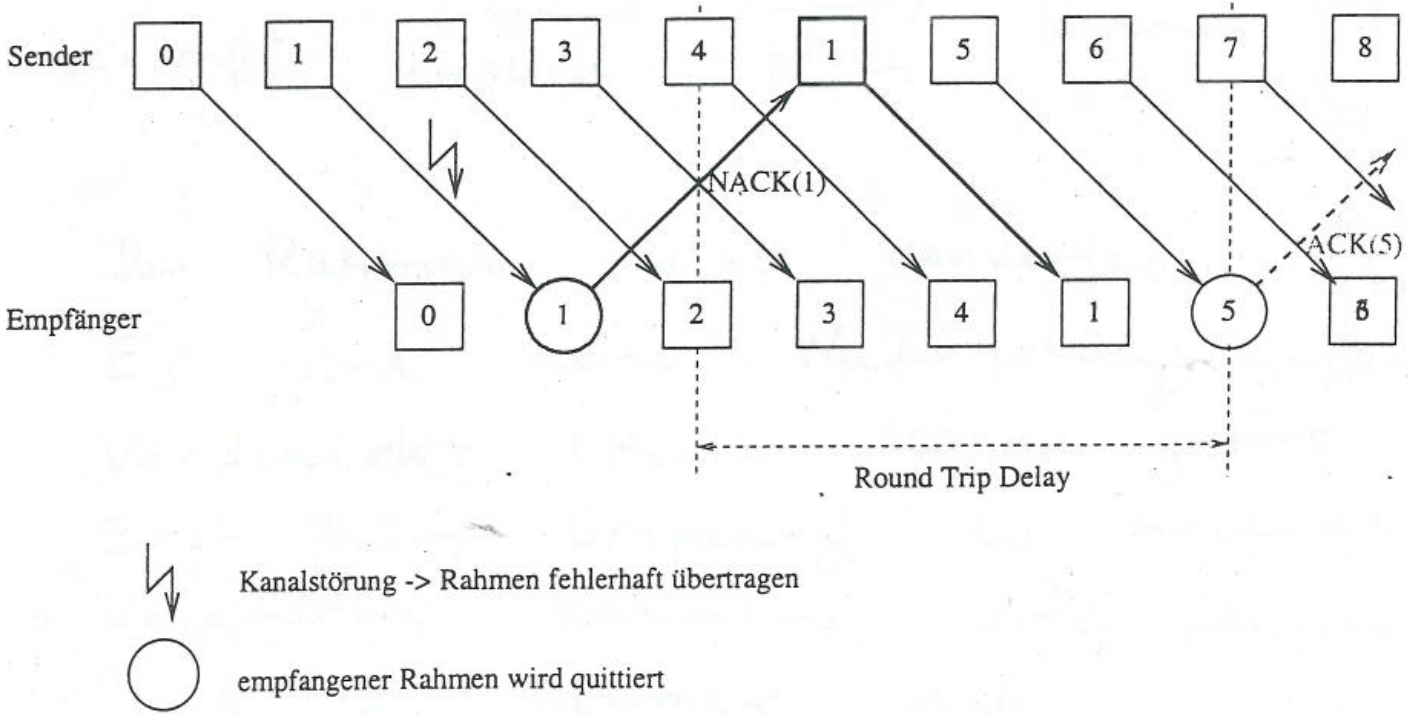
  Der Sender überträgt Rahmen k und folgende und gibt alle Rahmen bis $k-1$ frei.

- Verlorenes ACK/NACK

  Der Sender hat $n-1$ Rahmen gesendet und erwartet ein ACK/NACK, nach einer Zeitüberschreitung wird er alle Rahmen erneut senden.

Selective Repeat ARQ

- Empfänger hat Sliding Window.
- Sender kann einzelne Pakete wiederholen
- Out-of-Order Rahmen werden nicht verworfen.



Selective Repeat ARQ

Der Sender ist in der Lage, einzelne Rahmen erneut zu schicken, d.h er muß nicht nur die Nutzdaten, sondern die ganze Segmentierung speichern.

Der Empfänger quittiert zerstörte Rahmen oder fehlende Rahmen (zu erkennen an empfangenen Rahmen mit höherer Nummer) mit einem NACK(k). Weitere Rahmen $k+1, \ldots, k+l, l < n-1$ werden gespeichert.

Der Sender schickt Rahmen k erneut.

Der Empfänger quittiert alle Rahmen bis $k+l$ mit einem ACK($k+l+1$).

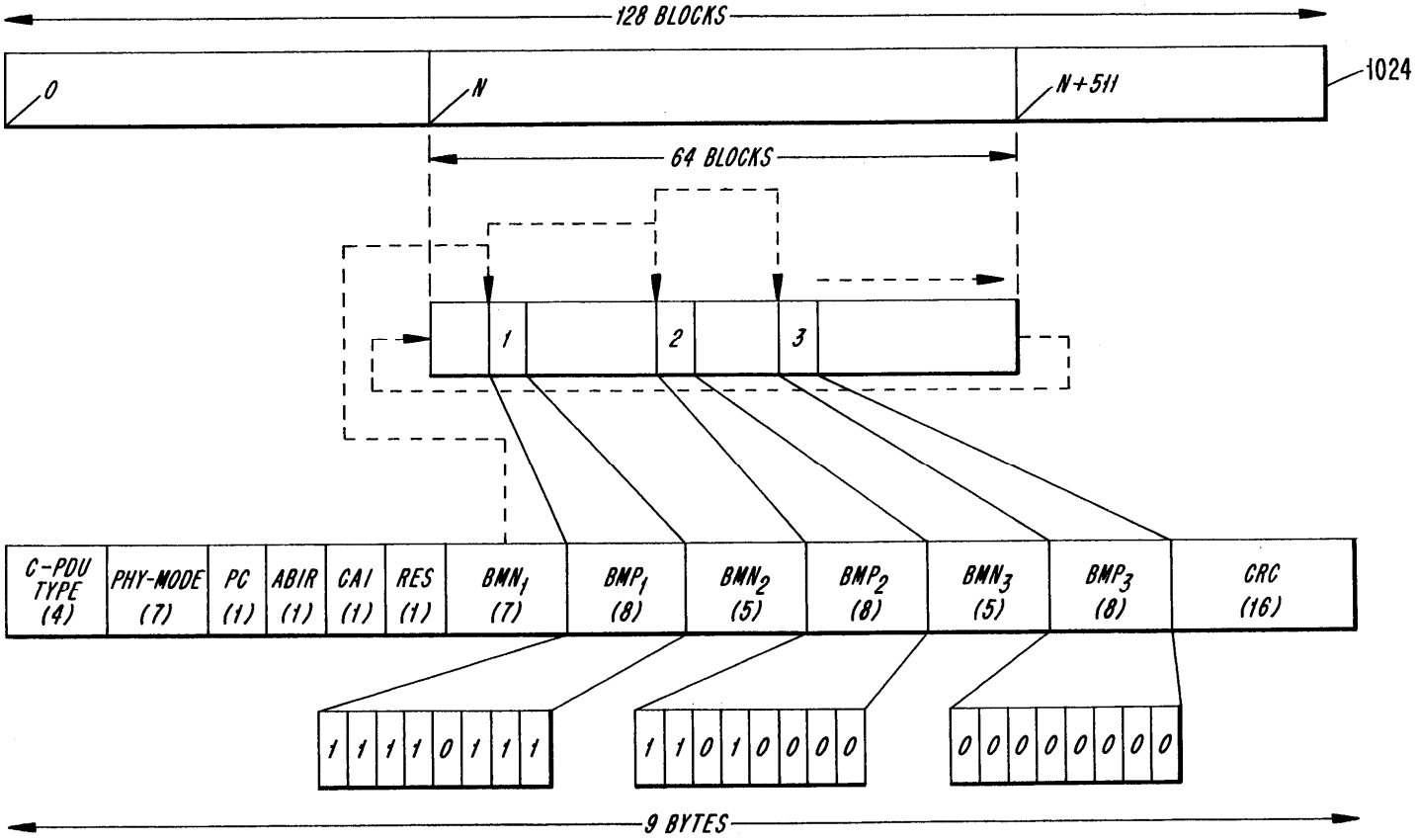
Bitmap-ARQ

Bitmap-ARQ ist eine effiziente Erweiterung von Selective Repeat ARQ

- Um Bandbreite bei den Kontrollpaketen (ACKs/NACKs) zu sparen, werden nicht einzelne ACKs/NACKs gesendet, sondern die ACK/NACK für eine Folge von Paketen werden in einem Bitmuster (Bitmap) codiert.
- Jedes Bit im Bitmuster steht für ein ACK/NACK
- Es können auch mehrere Bitmuster in einem Kontrollpaket gesendet werden.
- Für das erste Bitmuster wird die absolute Position im Buffer gesendet.
- Für folgende Bitmuster reicht die relative Position zum vorangehenden Bitmuster.

Bitmap-ARQ

- Beispiel mit drei Bitmustern
  (BMP = Bitmap, BMN = bitmap block number)

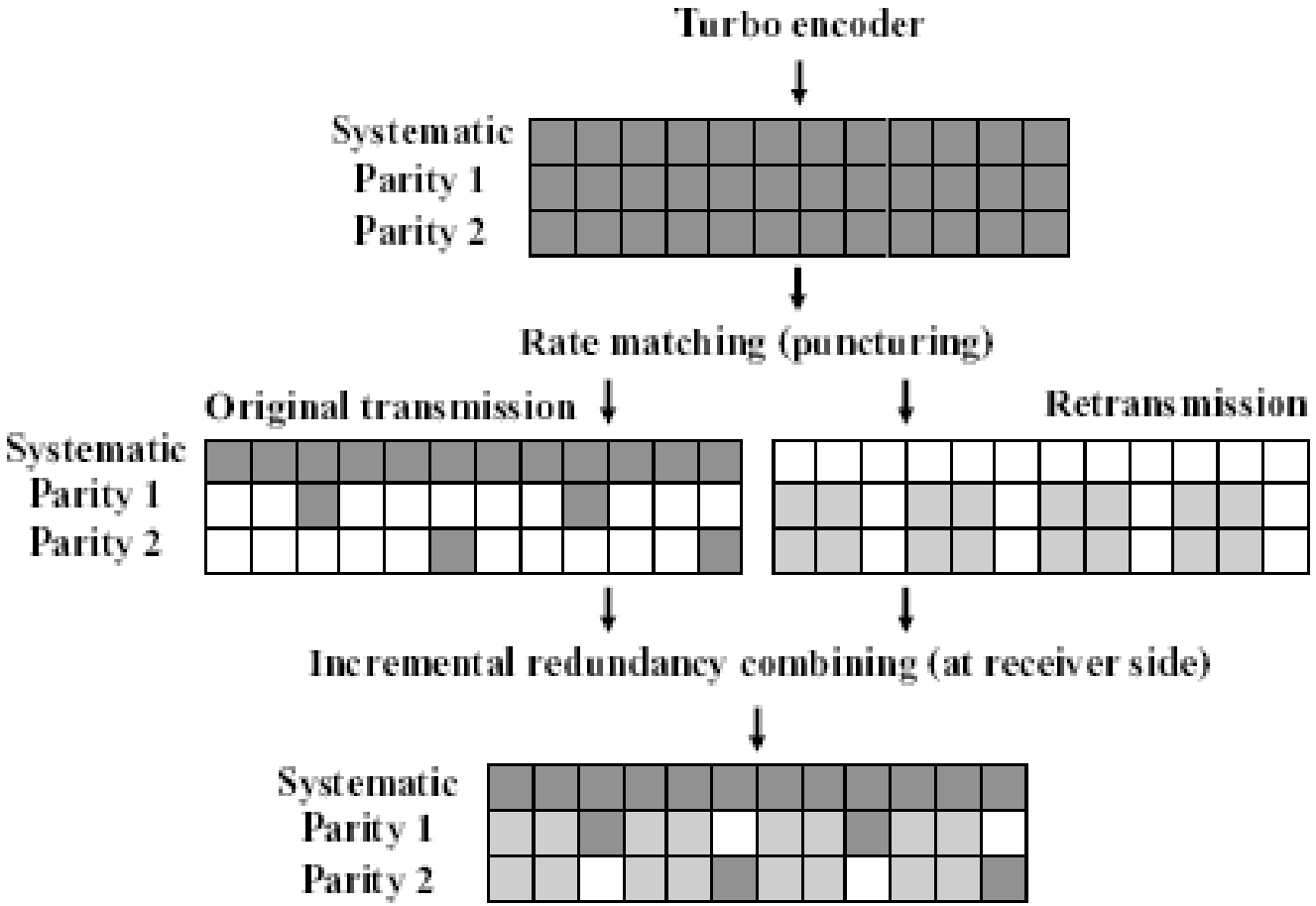


Hybrid ARQ

Hybrid ARQ kombiniert ARQ mit intelligenter Kanalcodierung

- Für ARQ kann z.B. Selective Repeat verwendet werden.
- Starke Kanalcodierung erzeugt ein große Menge an codierter Bits (z.B. 2 Parity-Bits je Daten-Bit bei Rate-1/3 Turbo Codierung).
- Im ersten Rahmen wird nur ein Teil der codierten Bits gesendet.
- In den Retransmissions werden jeweils andere codierte Bits übertragen (“incremental Redundancy”)
- Verbesserte Fehlerschutz durch Diversitätsgewinn durch mehr codierte Bits am Empfänger
- Verwendung z.B. bei UMTS/HSDPA (Das System ist sogar so ausgelegt, das im Normalfall der erste Rahmen nicht ausreicht)

Hybrid ARQ



(c) Holma and Toskala, WCDMA for UMTS

Mehrfachzugriffsverfahren (Multiple Access)

Motivation

Mehrere Stationen teilen sich ein physikalisches Übertragungsmedium, z.B.

- Frequenzbereich für Funkübertragung
- Kabel
- Lichtwellenleiter

Mögliche Verfahren, den Zugriff zu regeln, sind:

- Kanalpartitionierung (z.B. FDMA/TDMA/CDMA)
- Random Access
- Tokenverfahren (z.B. Token Ring, FDDI)

Beispiele für Kanalpartitionierung

- Time Division Multiple Access (TDMA)
- Frequency Division Multiple Access (FDMA)
- Code Division Multiple Access (CDMA)
- Space Division Multiple Access (SDMA)
- I/Q komplexe Modulation, z.B. QPSK, 16QAM

Beispielsysteme:

- GSM: TDMA, I/Q, (FDMA)
- WLAN: TDMA, I/Q, (FDMA)
- UMTS: CDMA, I/Q, (FDMA)
- Glasfaser: FDMA, (TDMA)

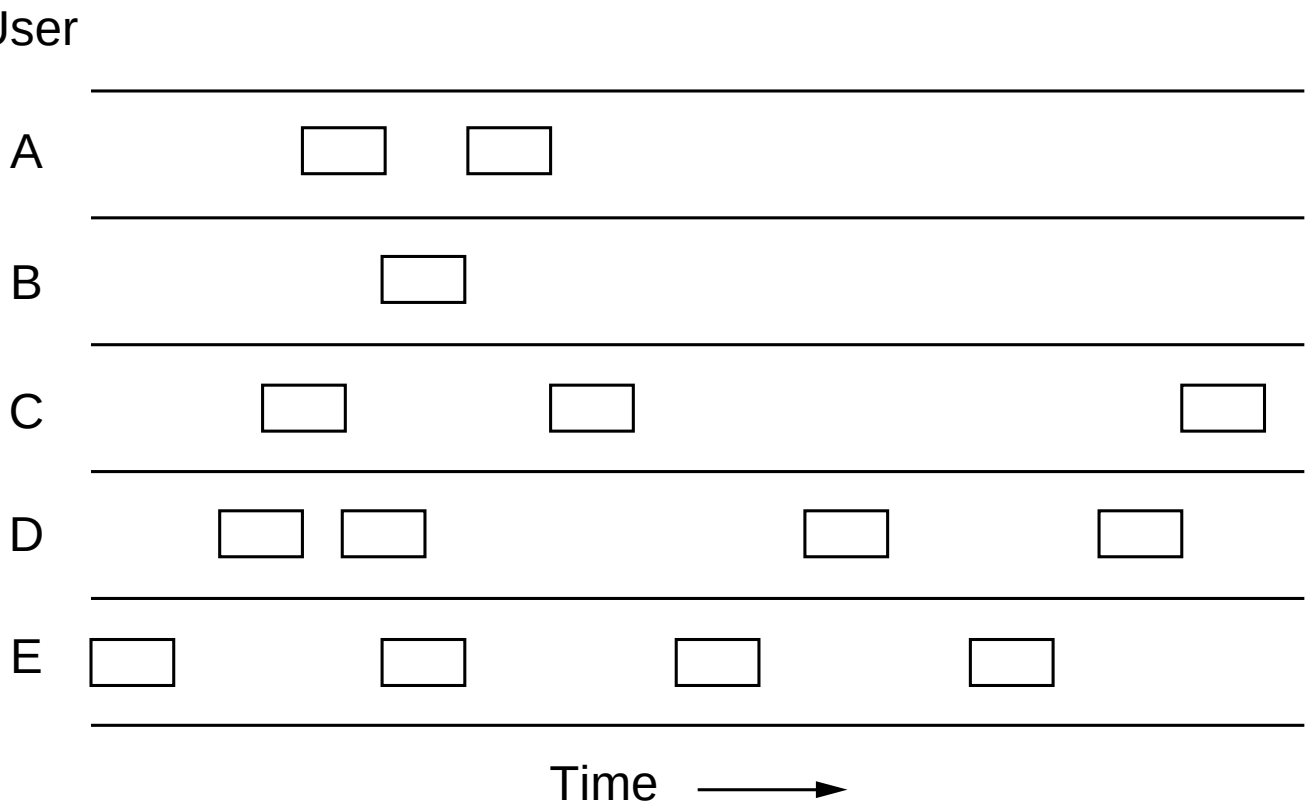
ALOHA Netz (Abramson, 1970)

- Jede Station überträgt generierte Rahmen sofort, ohne Rücksicht auf andere Stationen.
- Rahmen brauchen zur Übertragung eine feste Zeit T
- Die Station überwacht während des Sendens die Frequenz und erkennt so, ob auch eine andere Station gesendet hat und dadurch beide Rahmen zerstört wurden.
- Wird der Rahmen bei der Übertragung zerstört, wird er wiederholt:
  - Mit Wahrscheinlichkeit p wird er sofort neu übertragen.
  - Mit Wahrscheinlichkeit $1 - p$ wird nach Wartezeit T erneut geprüft, ob jetzt übertragen wird.

Vorteil des Verfahrens ist, daß keine Synchronsation zwischen den Stationen notwendig ist.

ALOHA

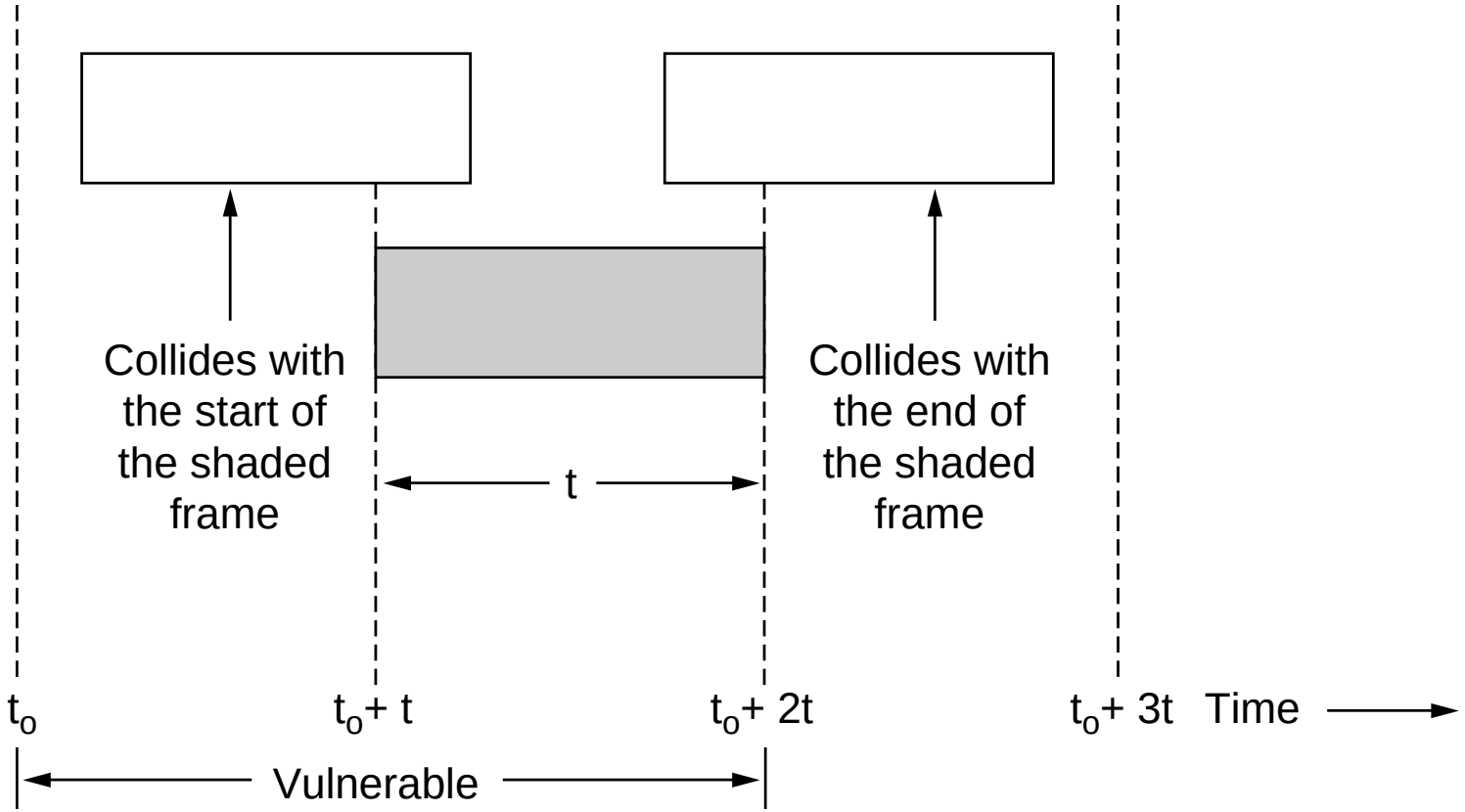
- Beim ALOHA Netz sind die Sendezeitpunkte zufällig
- Beispiel mit 5 Nutzern



(c) Tanenbaum, Computer Networks

ALOHA

- Kollision beim ALOHA Netz



(c) Tanenbaum, Computer Networks

Durchsatz des ALOHA Netzes

Rahmen starten zu Zeitpunkten $t_i, i \geq 0$.

Wir nehmen an, die Zeiten zwischen Rahmenanfängen sind stochastisch unabhängig, identisch verteilt mit Dichte $p(t) = \lambda\, e^{-\lambda\, t}$

Bezeichne N_x die Anzahl Rahmenstarts im einem Intervall der Länge x, dann genügt N_x einer Poissonverteilung mit Parameter $\lambda \cdot x$, d.h. die Wahrscheinlichkeit für genau k Rahmenstarts im Intervall der Länge x ist

$$P(N_x = k) = e^{-\lambda\, x}\, \frac{(\lambda\, x)^k}{k!}.$$

Durchsatz des ALOHA Netzes (2)

Erfolgreiche Übertragung gibt es dann, wenn im Intervall der Länge $2\,T$ genau ein Rahmen erzeugt wird, d.h. mit Wahrscheinlichkeit:

$$P(N_{2\,T} = 1) = 2\,\lambda\,T\,e^{-2\,\lambda\,T}$$

Das Maximum wird erreicht für $\lambda = \frac{1}{2\,T}$, d.h. wenn die mittlere Zeit zwischen zwei Rahmenanfängen $2\,T$ ist. Dann wird mit Wahrscheinlichkeit e^{-1} innerhalb von zwei Rahmenlängen genau ein Rahmen gesendet, was einer Kanalnutzung von ca. 18% entspricht.

Mit der mittleren Anzahl $G = \lambda T$ an Rahmen je Zeitschlitz T ergibt sich ein Durchsatz S

$$S = Ge^{-2G}$$

Slotted ALOHA

- Erweiterung von ALOHA bei der nur zu festgelegten diskreten Zeitpunkten gesendet werden darf.
- Erfordert Synchronisation zwischen den Sendern.
- Bei Kollisionen ist die Überlappung vollständig, d.h. die kritische Zeit in der kein weiterer Rahmen gesendet werden darf reduziert sich auf T.
- Verdoppelt die Kapazität von ALOHA.

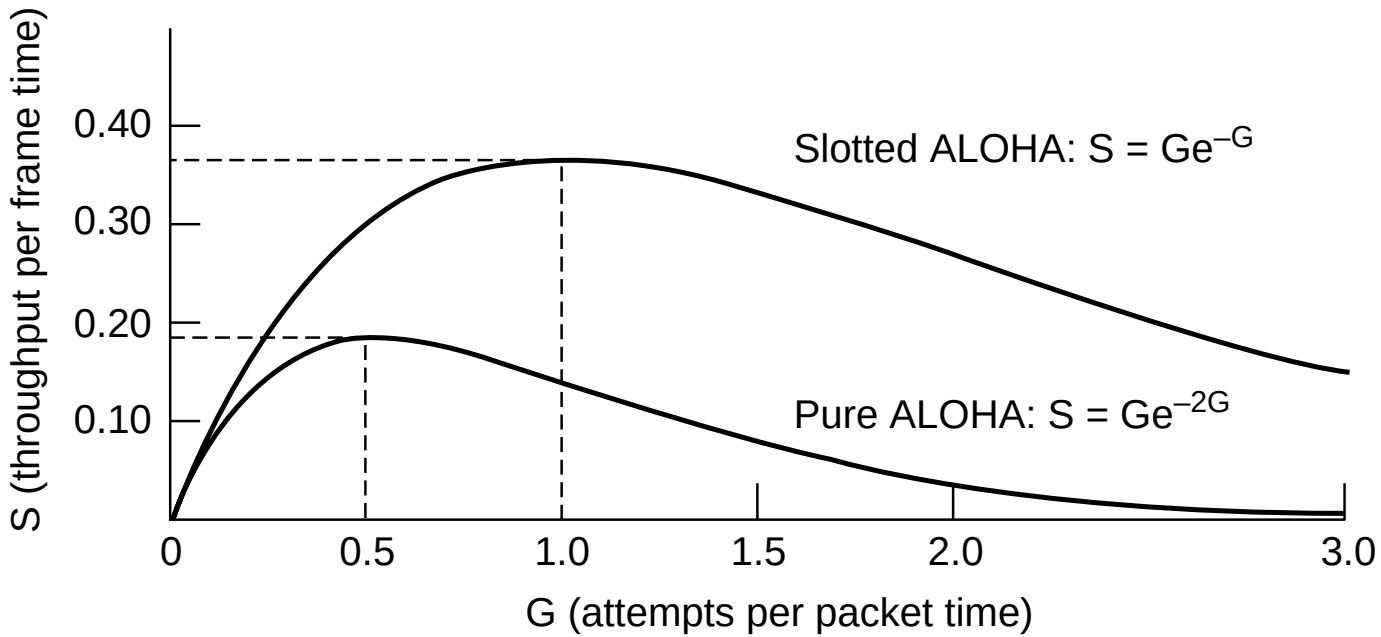
Als Durchsatz S ergibt sich daher

$$S = Ge^{-G}.$$

Das Maximum wird erreicht bei $G = 1$ bzw. $\lambda = 1/T$. Die Kanalnutzung ist dann ca. 37%.

Vergleich ALOHA und Slotted ALOHA

- Durchsatz von ALOHA und Slotted ALOHA



(c) Tanenbaum, Computer Networks

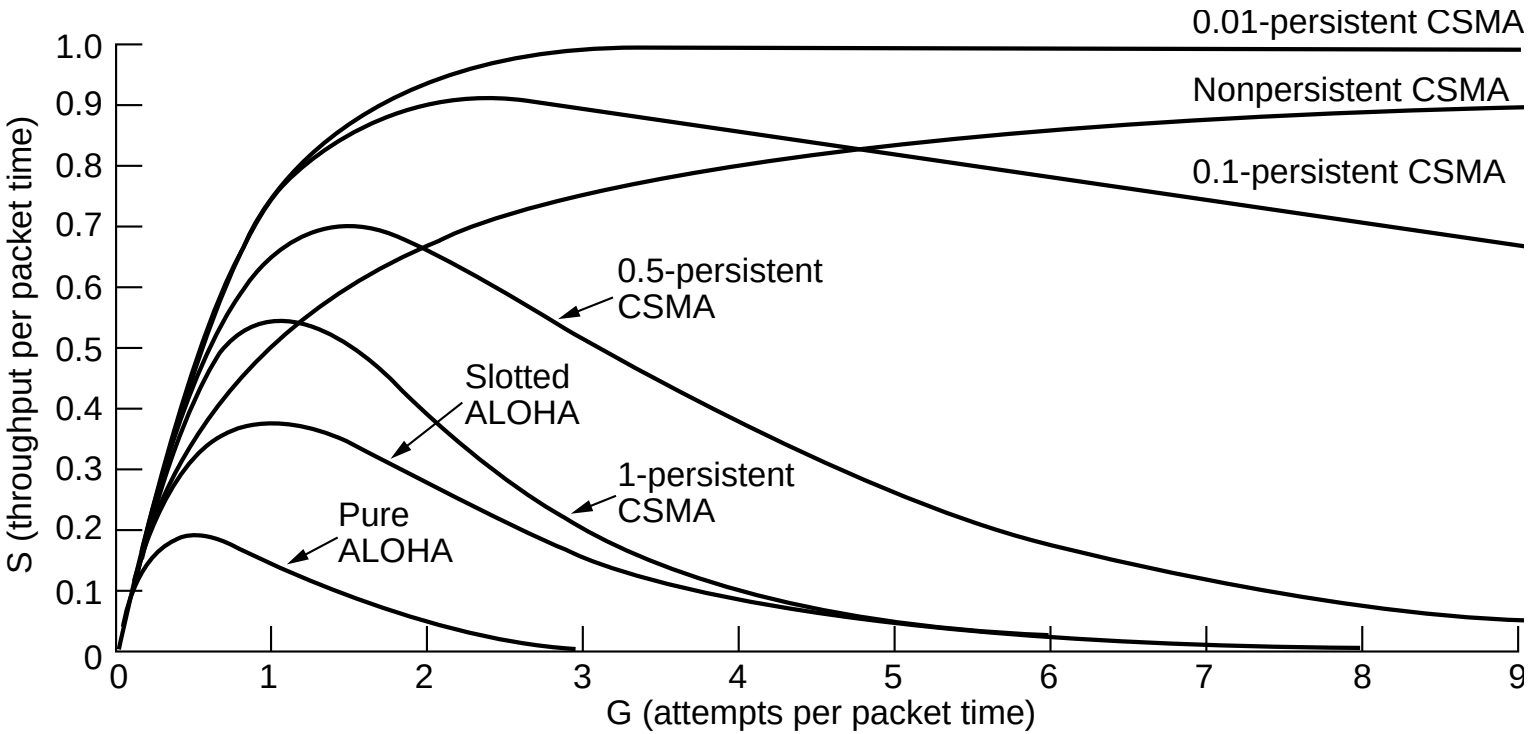
Carrier Sense Multiple Access (CSMA)

Im Gegensatz zu ALOHA überprüft bei CSMA jeder Sender ob der Kanal (bei ihm) gerade belegt ist (Carrier Sense) und sendet nur wenn der Kanal frei ist. Durch die Laufzeiten zwischen den Sendern kann es trotzdem zu Kollisionen kommen.

- persistant CSMA: Wenn ein Sender den Kanal wieder frei sieht, beginnt er direkt mit der Übertragung.
- non-persistant CSMA: Wenn ein Sender den Kanal wieder frei sieht, wartet er eine zufällige Zeit bis zur Übertragung.
- *p*-persistant CSMA: Wenn ein Sender den Kanal wieder frei sieht, sendet er direkt mit der Wahrscheinlichkeit p.

Vergleich ALOHA und CSMA

- Durchsatz von CSMA und (Slotted) ALOHA



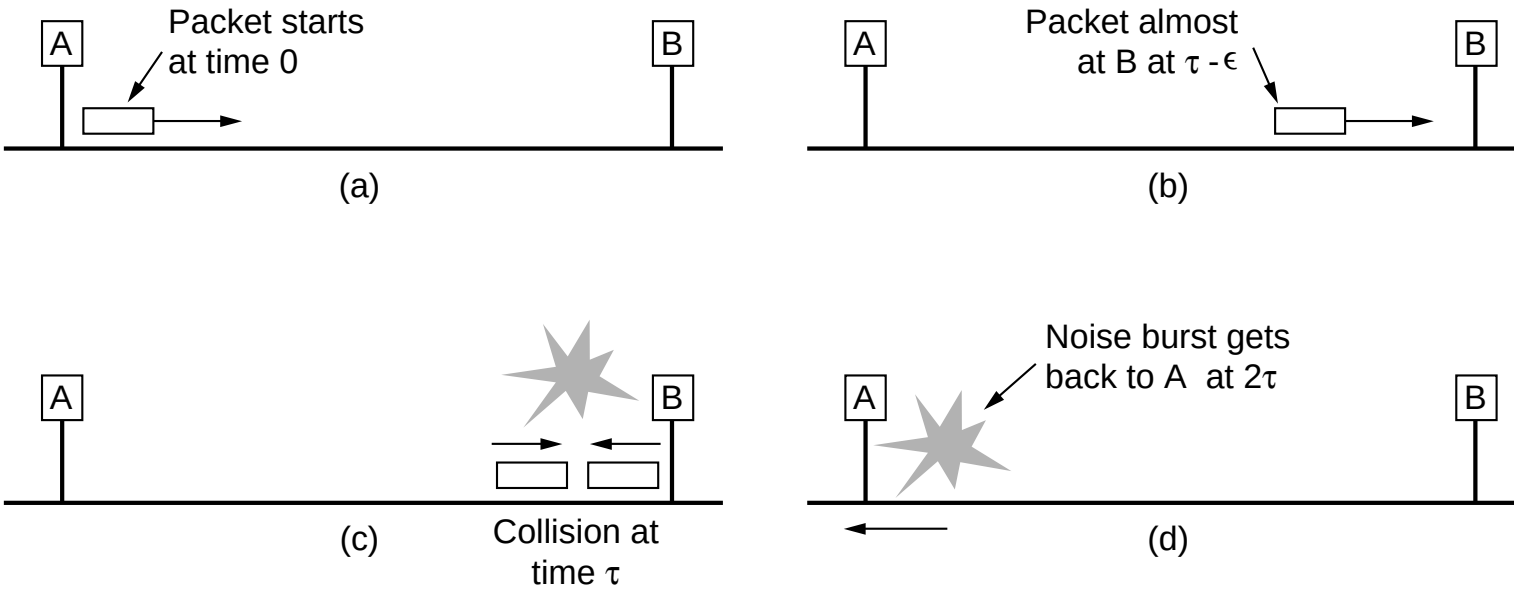
(c) Tanenbaum, Computer Networks

Ethernet mit CSMA mit Collision Detection (1)

Ethernet benutzt folgendes Verfahren (CSMA/CD: Carrier Sense Multiple Access / Collision Detection)

- Bevor ein Rahmen gesendet wird, wartet der Ethernet Adapter, daß der Link für die Zeit von 96 Bit frei ist.
- Während des Sendens prüft der Adapter, daß keine andere Station sendet.
- Erkennt er das Signal einer anderen Station, sendet er ein 48 Bit langes Störsignal und bricht die Übertragung ab.
- Wurde ein Rahmen durch Kollisionen schon n mal zerstört, wartet der Sender eine zufällige Zeiteinheiten von 512 Bit, die zwischen 0 und $2^{\min\{n,10\}} - 1$ liegt.
- Mindestgröße für Rahmen ist 512 Bit (64 Byte) zur effektiven Rechtzeitigen Kollisionserkennung

Ethernet mit CSMA mit Collision Detection (2)



(c) Tanenbaum, Computer Networks

- Übertragungszeit t_{trans}: Sendedauer für einen Rahmen
- Ausbreitungszeit t_{prop} (oder τ): Laufzeit eines Signal von Sender zum Empfänger
- Zur sicheren Kollisionserkennung sollte gelten: $t_{trans} > 2t_{prop}$.

Beispiele

Einführung Point-To-Point Protocol

Das Point-to-Point Protocol (PPP, vgl. RFC1661, RFC1662, RFC1663) bietet eine exklusive Verbindung zwischen zwei Netzwerkknoten.
Anwendungen sind:

- Anbindung von Rechnern zum ISP
- Verbindung zwischen Routern

Für Designaspekte und Anforderungen siehe RFC1547. Nicht angeboten werden:

- Flußkontrolle
- Fehlerbehebung
- Mehrfachzugriff

Rahmenbildung gemäß RFC1662

Flag	Address	Control	Protocol	Info	Checksum	Flag

Flag **01111110**, Byte Stuffing mit Escape **01111101**, nächstes Byte wird XOR 20_{16} =**00100000** gesendet.

Address Immer **11111111**

Control Immer **00000011**

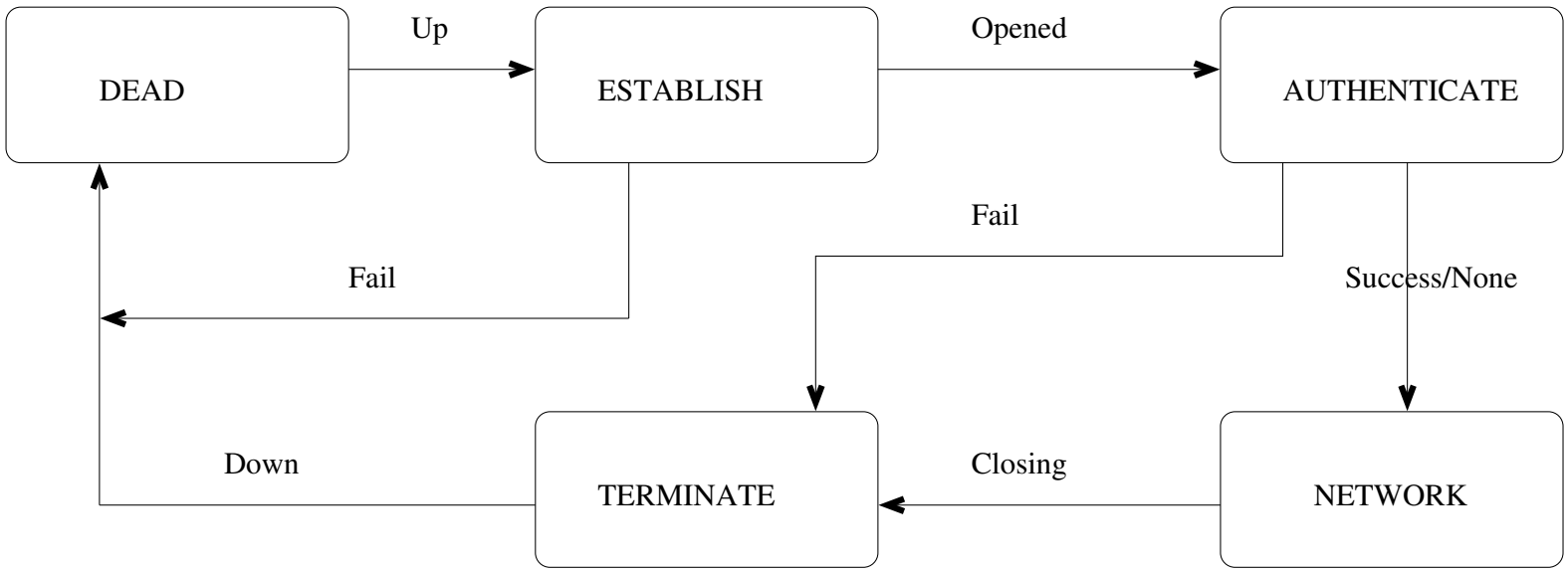
Protocol 16Bit Protokollspezifikation gemäß RFC1700, PPP DLL Protocol Numbers

Info Variable Anzahl Daten, gemäß RFC1547 sollen mindestens 1500Byte möglich sein.

Check Üblicherweise 16Bit Prüfsumme mit dem CRC-CCITT Generatorpolynom, auch 32Bit mit Ethernet Generatorpolynom ist möglich (vgl. RFC1662)

Flag Wie am Rahmenanfang: **01111110**

Zustandsgraph



Zustandsübergänge

- **DEAD:** Startzustand
- **ESTABLISH:** Link Control Protocol (LCP, Protocol $C021_{16}$) konfiguriert den Link, andere Pakete werden in diesem Zustand verworfen.
- **AUTHENTICATE:** Authentifizierung der Gegenseite, falls erforderlich. Übliche Protokolle sind PAP (RFC1334), CHAP (RFC 1994), EAP (RFC3748).
- **NETWORK:** Die Netzwerkschicht benutzt ihr Network Control Protocol, um den Link zu konfigurieren. Dann können die Netzwerkschichten den Link nutzen.
- **TERMINATE:** LCP Terminate Pakete dienen dazu, die Verbindung zu schließen.

Einführung Ethernet

Ethernet wird üblicherweise für Lokale Netze (LANs) verwendet. Als physikalisches Medium dienen Kupferkabel oder Lichtwellenleiter.

- Initiale Entwicklung durch Robert Metcalfe Anfang der 70er Jahre für Xerox
- Standardisiert 1982 durch Digital Equipment/Intel/Xerox
- Inkompatibel standardisiert als IEEE Standard 802.3, parallel zu 802.4 (Token Bus von General Motors) und 802.5 (Token Ring von IBM)
- Heute standardisiert bis 10Gbit pro Sekunde.
- 100Gbit befindet sich in der Entwicklung

Rahmen

Preamble	SOF	Destination	Source	Length/Type	Info	Padbytes	Checksum

- **Preamble/SOF:** 7 Byte zur Synchronisation **10101010**, dann Start Of Frame Byte.
- **Destination/Source:** Adressen von Zieladapter und Quelladapter, jeweils 6 Byte
- **Length/Type:** 2 Byte Typ der gekapselten Daten benutzt. Werte gemäß RFC1700, Ether Types. Der IEEE Standard erlaubt hier die Länge der Daten.
- **Info:** 1-1500 Bytes Nutzdaten, die Länge muß aus den Daten für höhere Schichten ersichtlich sein.
- **Padbytes:** Info und Padbytes müssen zusammen mindestens 46 Byte enthalten.
- **Checksum:** CRC-32 Prüfsumme

Ethernet Adressen

Schreibweise ist üblicherweise 6 Bytes hexadezimal, getrennt durch Doppelpunkt, z.B. $00:08:02:6D:DB:66$.

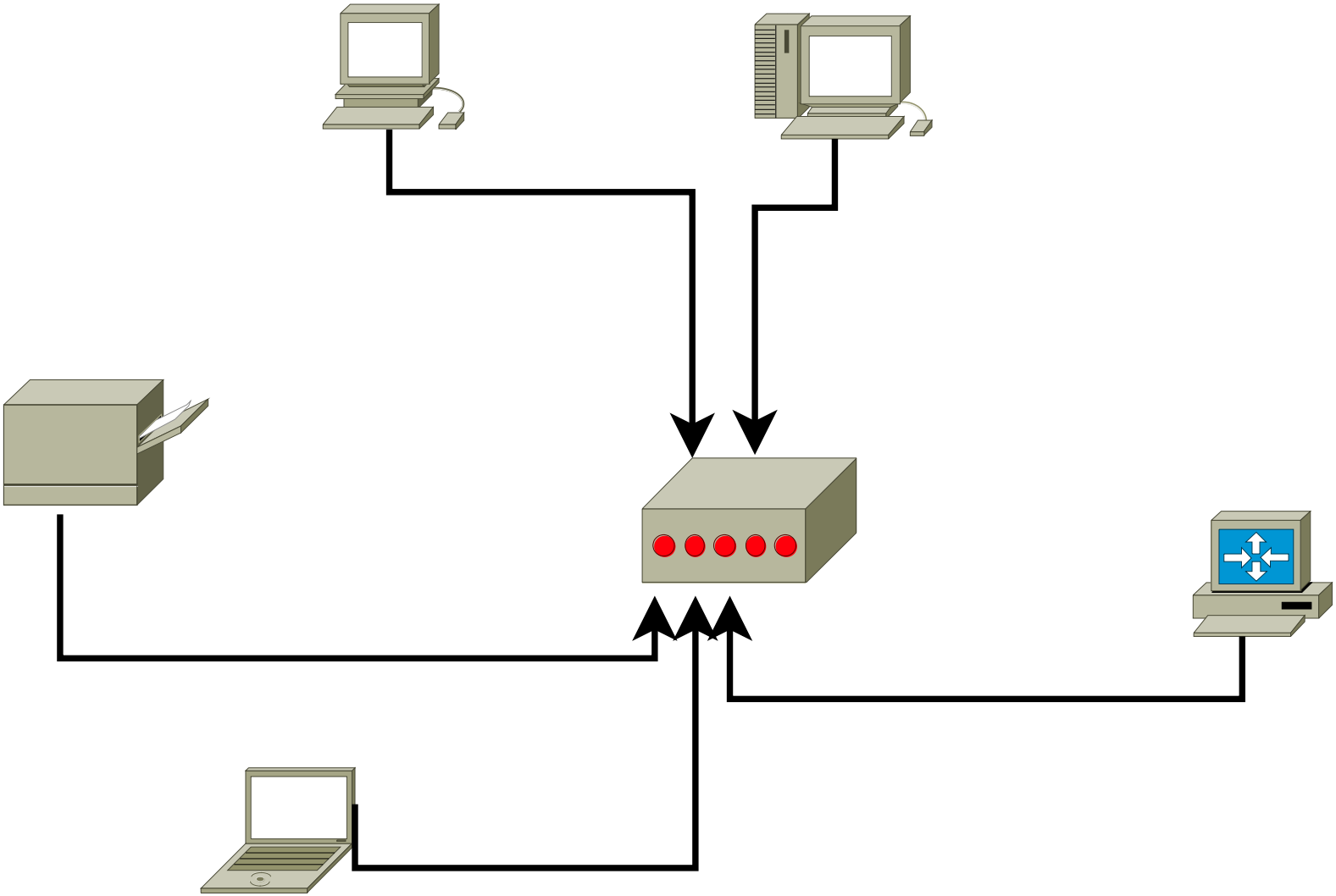
- Die ersten 3 Byte sind einem Hersteller zugewiesen, Verwaltet als Organizationally Unique Identifier (OUI) durch IEEE
  z.B. $00:08:02$ für Compaq (heute HP)
- Die letzten 3 Byte bilden die Seriennummer.
- $FF:FF:FF:FF:FF:FF$ ist die Broadcast Adresse.
- Die zwei niederwertigen Bit von Byte 1 der Adresse haben spezielle Bedeutung:
  - Bit 0 = 0: Unicast Adresse
  - Bit 0 = 1: Multicast Adresse/Broadcast Adresse
    z.B. $0\mathbf{3}:00:00:20:00:00$ IP Multicast, RFC1469
  - Bit 1 = 0: OUI Adresse
  - Bit 1 = 1: Lokal administrierte Adresse

Ethernet Switches

Heutige LANs auf Ethernet-Basis benutzen in der Regel eine Topologie, bei der jeder Endpunkt mit einem Ethernet Switch verbunden ist. Switche sind wiederum mit weiteren Switchen oder Routern verbunden.

- Mehrfachzugriff spielt keine Rolle, da jeder Link von zwei Endpunkten im Duplexverfahren benutzt werden kann.
- Unterschiedliche Endgeräte können unterschiedliche Bitübertragungsschichten nutzen (z.B. Glasfaser/Kupfer unterschiedlicher Geschwindigkeit)
- Pakete sind nur auf den Links sichtbar, die sie benötigen.

Ethernet Switches

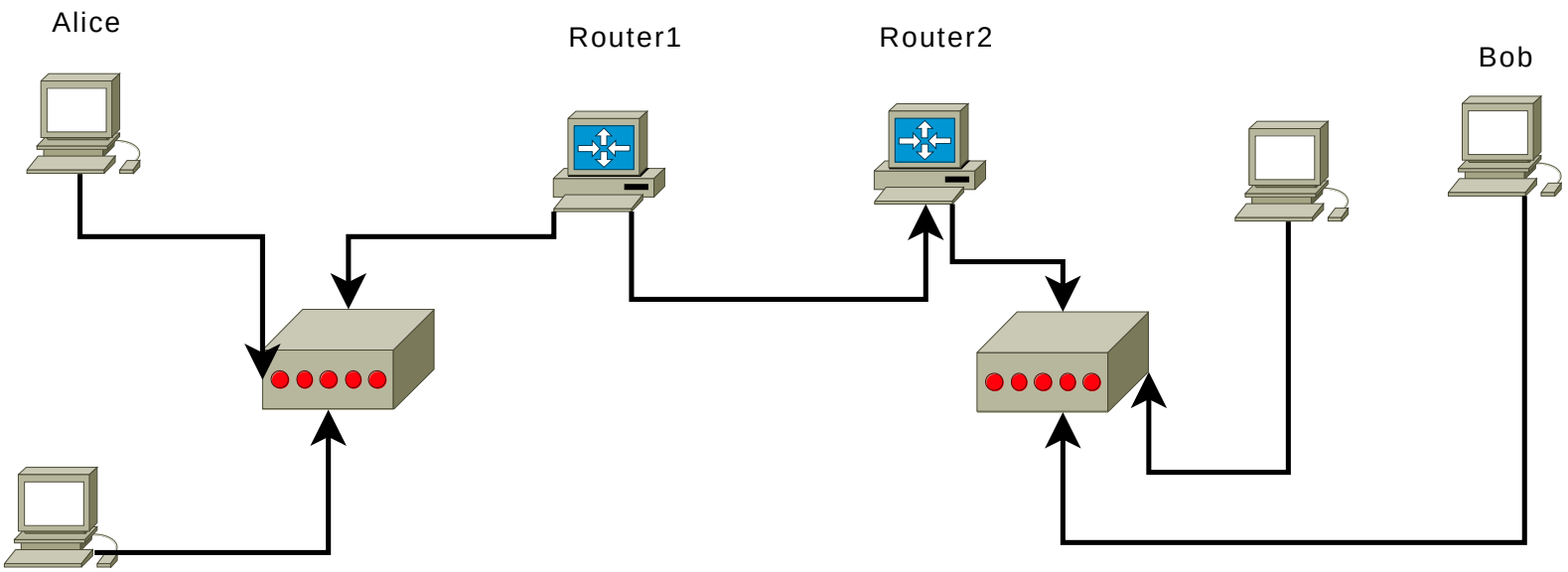


Weiterleitung von Paketen

Jeder Switch speichert zu jedem Port eine Tabelle der Ethernet Adressen, die hinter diesem Port als Quelladresse verwendet wurden.

- Hat der Switch die Zieladresse in der Tabelle eines Ports, leitet er das Paket nur auf diesem Port weiter.
- Kennt der Switch die Adresse nicht, leitet er das Paket an alle (benutzten) Ports weiter.
- Broadcast und Multicast Adressen sind nie Quelladresse.

Address Resolution Protocol: Motivation



Wir betrachten die Kommunikationsbeziehung von Alice zu Bob

Die Netzwerkschicht kennt die Netzwerkadresse des Zielrechners und kann damit die Netzwerkadresse des nächsten, d.h. mit einer Adresse der Sicherungsschicht erreichbaren, Knotens (Router1) bestimmen.

Aufgabe der Sicherungsschicht ist es, die Netzwerkadresse des nächsten Knotens in eine brauchbare Adresse umzuwandeln.

- Statische Konfiguration
  (z.B. Tabelle IP <-> Ethernet Adresse)
- Dynamisch mittels Abfrage im lokalen Netz

Rahmenstruktur, vgl. RFC826

Destination	Source	Type	Hard Type	Prot Type	Hard Size	Prot Size	Op	Sender MAC	Sender Network	Target MAC	Target Network
6	6	2	2	2	1	1	2				

- **Destination:** Ethernet Zieladresse
- **Source:** Ethernet Quelladresse
- **Type:** ARP, d.h. 0806_{16}, vgl. RFC1700

Felder im Rahmen

- **Hardware Type:** Art der nachgefragten Adresse, z.B. 1 für Ethernet
- **Protocol Type:** Art der Netzwerkadresse, z.B. 0800_{16} für IP, vgl. RFC1700
- **Hardware Size, Protocol Size:** Länge der Adressen, 6 für Ethernet, 4 für IPv4
- **Operation:** Typ des Requests
  1. ARP Request, d.h. Netzwerk $\rightarrow$ Hardware
  2. ARP Reply, Antwort
  3. RARP Request, d.h. Hardware $\rightarrow$ Netzwerk
  4. RARP Reply, Antwort dazu.
- **Sender MAC:** Identisch mit **Source**
- **Sender Network:** Netzwerkadresse der Quelle
- **Target MAC:** Hardwareadresse des Ziels
- **Target Network:** Netzwerkadresse der Ziels

Funktionsweise

Wann immer zu einer Netzwerkadresse die passende Adresse der Sicherungsschicht bestimmt werder muß, wird ein ARP Request mit entsprechenden Daten erzeugt und an alle Hosts des lokalen Netzes dieses Adapters geschickt.

Empfängt der Host mit passender Netzwerkadresse den ARP Request, antwortet er mit einem ARP Reply, in dem er seine Hardwareadresse als **Sender MAC** einsetzt. Das Ergebnis bleibt eine bestimmte (üblicherweise konfigurierbare) Zeit gespeichert (ARP Cache)

Wird der Request nicht beantwortet, wird das Verfahren nach wenigen Wiederholungen abgebrochen.

Empfängt ein Host einen ARP Request von einem Host, dessen IP im ARP Cache ist, wird die Quelladresse automatisch übernommen.

Beispiel ARP Request

```
Ethernet II
  Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  Source: Agere_66:79:ca (00:02:2d:66:79:ca)
  Type: ARP (0x0806)
Address Resolution Protocol (request)
  Hardware type: Ethernet (0x0001)
  Protocol type: IP (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (0x0001)
  Sender MAC address: Agere_66:79:ca
  Sender IP address: 134.61.33.148
  Target MAC address: 00:00:00_00:00:00
  Target IP address: 134.61.32.1
```

Beispiel ARP Reply

```
Ethernet II
  Dest.: Agere_66:79:ca (00:02:2d:66:79:ca)
  Source: Ibm_3e:81:76 (00:14:5e:3e:81:76)
  Type: ARP (0x0806)
Address Resolution Protocol (reply)
  Hardware type: Ethernet (0x0001)
  Protocol type: IP (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: reply (0x0002)
  Sender MAC address: Ibm_3e:81:76
  Sender IP address: 134.61.32.1
  Target MAC address: Agere_66:79:ca
  Target IP address: 134.61.33.148
```

Gratuitous ARP

Von **Gratuitous ARP** spricht man, wenn ein Host mit einem ARP Request im LAN nach seiner eigenen Hardwareadresse fragt. Dies kann zwei Gründe haben:

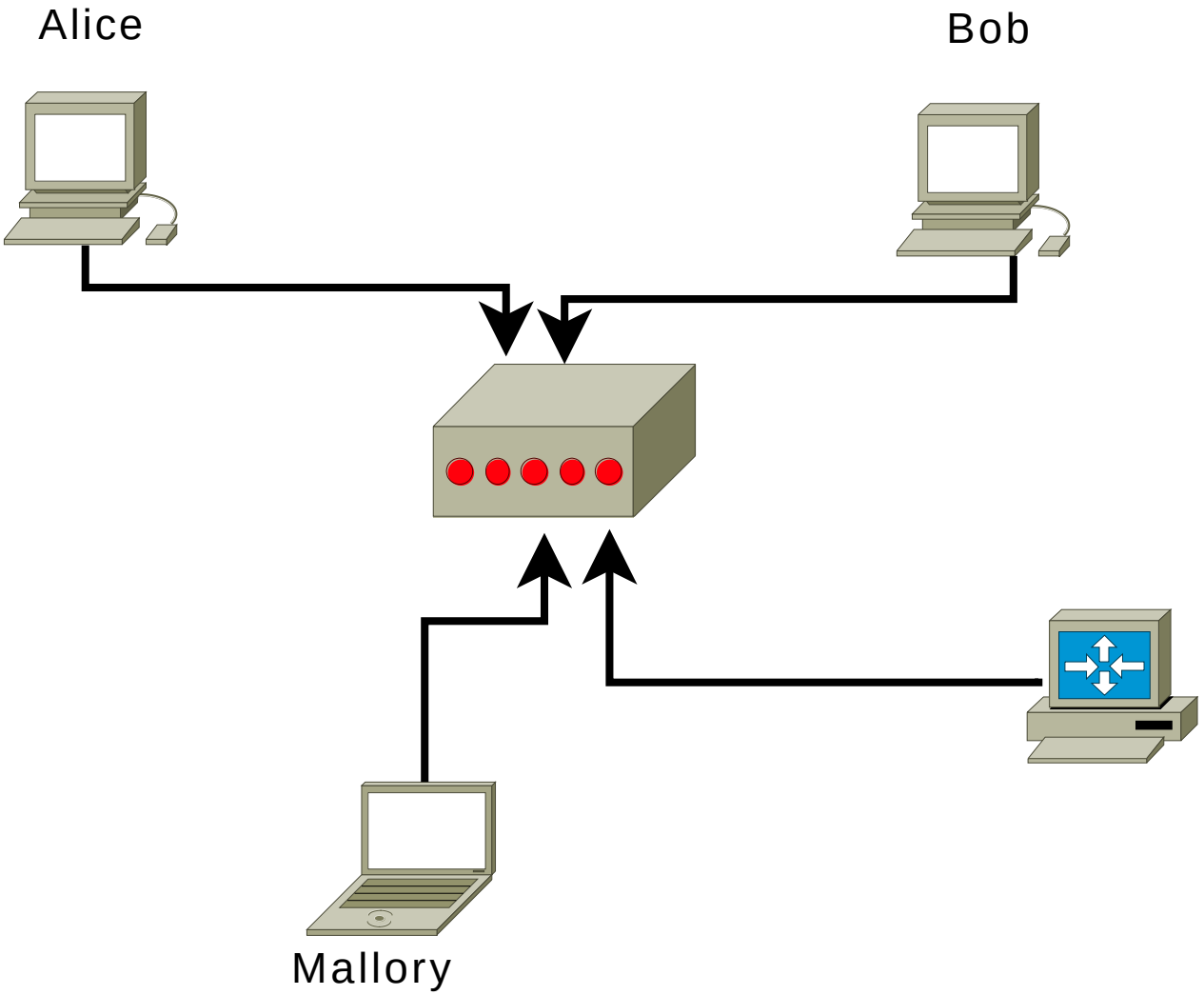
1. Wird der Request beantwortet, hat ein weiterer Host im LAN dieselbe Netzwerkadresse, was auf eine Fehlkonfiguration hindeutet und den Netzbetrieb stören kann.
2. Alle Hosts, die schon einen Eintrag zu dieser Netzwerkadresse im ARP Cache haben, werden über die (gegebenenfalls andere) Hardwareadresse informiert.

Grundlage ARP Poisoning

Falls ein Ethernet Switch die Quell-MAC eines Rechners kennt, leitet er Pakete an diesen Rechner nur noch auf dem Port weiter, hinter dem der Rechner erreichbar ist.

Möchte ein Angreifer die Kommunikation zweier anderer Rechner belauschen, kann er dazu den Switch oder die Endpunkte angreifen.

Beispielszenario



Trivialansatz

Manche Ethernet Switche haben eine (natürlich endliche) Tabelle mit bekannten Quelladressen. Gelingt es Mallory, durch Ethernet Pakete mit falschen Quelladressen, die MAC von Bob aus der Tabelle zu spülen, werden Pakete von Alice an Bob auf allen Ports sichtbar.

Der Ansatz funktioniert in der Regel nicht, da die Antwortpakete von Bob das Problem auf dem Switch sofort wieder beheben.

Beispielprogramm: macof

ARP Spoofing

Mallory sendet ARP Reply Pakete an Alice und Bob, in denen für die Netzwerkadresse vom jeweils anderen Host die Ethernetadresse von Mallory angegeben wird.

Alice und Bob werden mit diesen Paketen den ARP Cache auffrischen und daher Pakete an die Netzwerkadresse des Kommunikationspartners an den Ethernet Adapter von Mallory senden.

Mallory späht die Pakete aus und leitet sie an den echten Empfänger weiter.

Beispielprogramme: arpspoof, ettercap
Verteidigung: arpwatch
vgl. §202c StGB

Vermittlungsschicht

Funktionen der Vermittlungsschicht

Aufgabe der Vermittlungsschicht ist es, Daten von einem Knoten zu einem (oder mehreren) Knoten des Netzes zu übertragen. Dabei können sowohl Switche als auch Router (Knoten, die die Weiterleitungsentscheidung anhand der Netzwerkadresse treffen) im Pfad liegen.

Typische Funktionen der Vermittlungsschicht sind

- **Weiterleitung** eines Paketes an den nächsten Knoten
- **Routing** eines Paketes von der Quelle zum Ziel
- **Verbindungsaufbau**, falls das Netzwerk diese Funktion erfordert.

mögliche Dienste

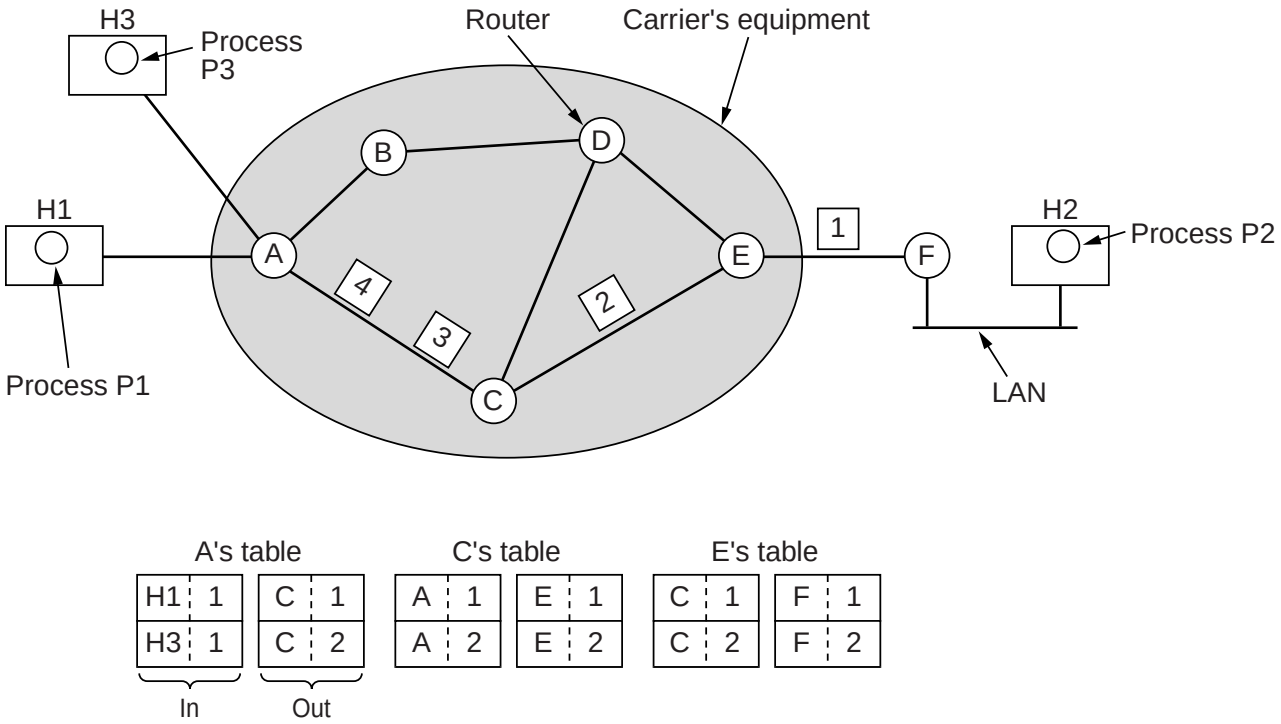
Die Vermittlungsschicht kann der Transportschicht eine Vielzahl von Diensten anbieten, darunter:

- Garantierte Auslieferung: Die Pakete erreichen ihr Ziel oder der Sender erhält eine Fehlermeldung.
- Garantierte Auslieferung in garantierter Zeit: Auch die Zeit bis zur Ankunft am Ziel ist garantiert.
- Garantierter Reihenfolge: Die Pakete kommen in der Reihenfolge an, in der sie gesendet wurden.
- Garantiertes Verzögerungsintervall: Die Auslieferzeit variiert nur in einem gegebenen Intervall.
- Sicherheitsdienste: Transparente gegenseitige Authentifizierung, Nachrichtenintegrität und Vertraulichkeit.

Leitungsvermittlung / (Virtual) Circuit Switching

- **Verbindungsaufbau:** Es wird zuerst eine Ende-zu-Ende Verbindung hergestellt. Diese kann Dienstgütevereinbarungen enthalten, die von jedem beteiligten Knoten einzuhalten sind.
- **Datenübertragung:** Zwischen je zwei Knoten besteht eine virtuelle Verbindung mit einer bestimmten Kennung. In Paketen wird von Routern jeweils die Kennung der Verbindung zum nächsten Knoten eingesetzt.
- **Verbindungsabbau:** Die zu einer Verbindung gehörenden Einträge der Routingtabellen entlang des Pfades werden gelöscht.

Leitungsvermittlung / (Virtual) Circuit Switching



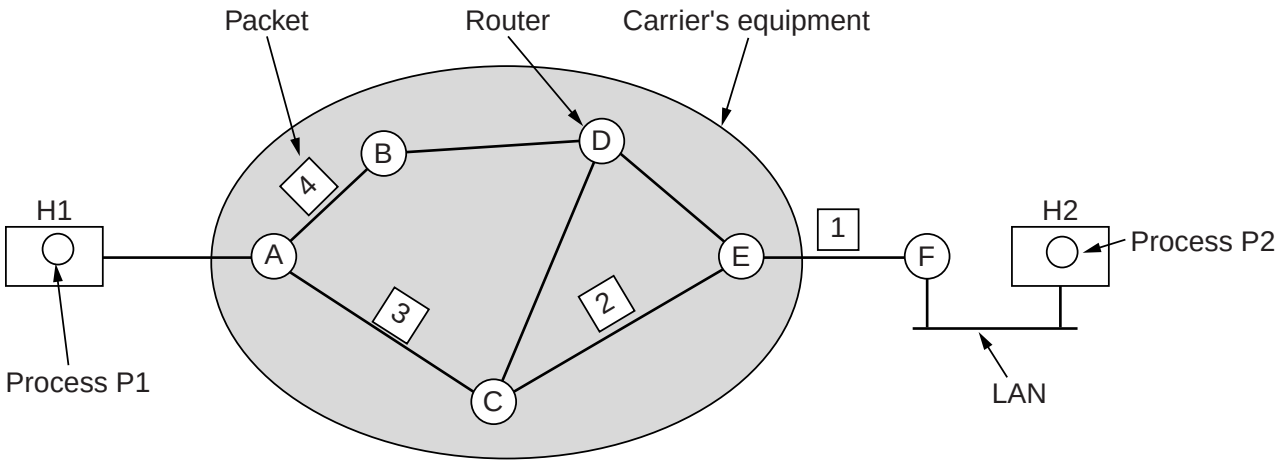
(c) Tanenbaum, Computer Networks

Routing Tabelle enthält Port und Verbindungsnummer.

Paketvermittlung / Paket Switching

- **Verbindungsaufbau:** Gibt es nicht.
- **Datenübertragung:** Jedes Paket enthält den vollständigen Adresssatz und wird anhand der Zieladresse unabhängig von anderen Paketen weitergeleitet. Der Router stellt anhand einer Tabelle der Zieladressen fest, über welches Ausgangsinterface ein Paket weitergeleitet wird.
- **Verbindungsabbau:** Gibt es nicht.

Paketvermittlung / Paket Switching



A's table

initially		later	
A	–	A	–
B	B	B	B
C	C	C	C
D	B	D	B
E	C	E	B
F	C	F	B

Dest. Line

C's table

A	A
B	A
C	–
D	D
E	E
F	E

E's table

A	C
B	D
C	C
D	D
E	–
F	F

(c) Tanenbaum, Computer Networks

Geschichte und Vergleich

- **Leitungsvermittlung** stammt aus der Telefonwelt
  - Endgeräte können extrem dumm sein.
  - Dienstgüteanforderungen sind leicht einzuhalten.
  - Kontrolle durch Diensteanbieter ist einfach.
  - Beispiele: POTS (plain old telephone service), ATM (asynchronous transfer mode)
- **Paketvermittlung** wurde zur robusten Verbindung von Rechnersystemen geschaffen
  - Endgeräte müssen ggfs. aus Paketen den Datenstrom rekonstruieren.
  - Dienstgüte ist kaum zu garantieren.
  - Leicht erweiterbar, da keine Dienstgüteanforderungen und wenig Kontrolle
  - Beispiel: Internet