Dienste der Sicherungsschicht

- Unbestätigter, verbindungsloser Dienst
- Bestätigter, verbindungsloser Dienst
- Betsätigter, verbindungsorientierter Dienst

Methoden in der Sicherungsschicht

- Rahmenbildung
- Fehlererkennung und Fehlerkorrektur
- Flußkontrolle
- Mehrfachzugriff (Multiple Access)

Methoden zur Rahmenbegrenzung

- Zeichenzahl
  In der Sicherungsschicht allein nur noch sehr selten verwendet, da Übertragungsfehler der Rahmenlänge nicht ohne andere Verfahren erkannt bzw. korrigiert werden können.
- Flagbytes
- Start- und Endflag
- Kodierungsverletzungen
  Das Verfahren benutzt Redundanz in der Bitübertragungsschicht.
  Beispiel: 100base-TX verwendet 11000 10001

Flag Bytes

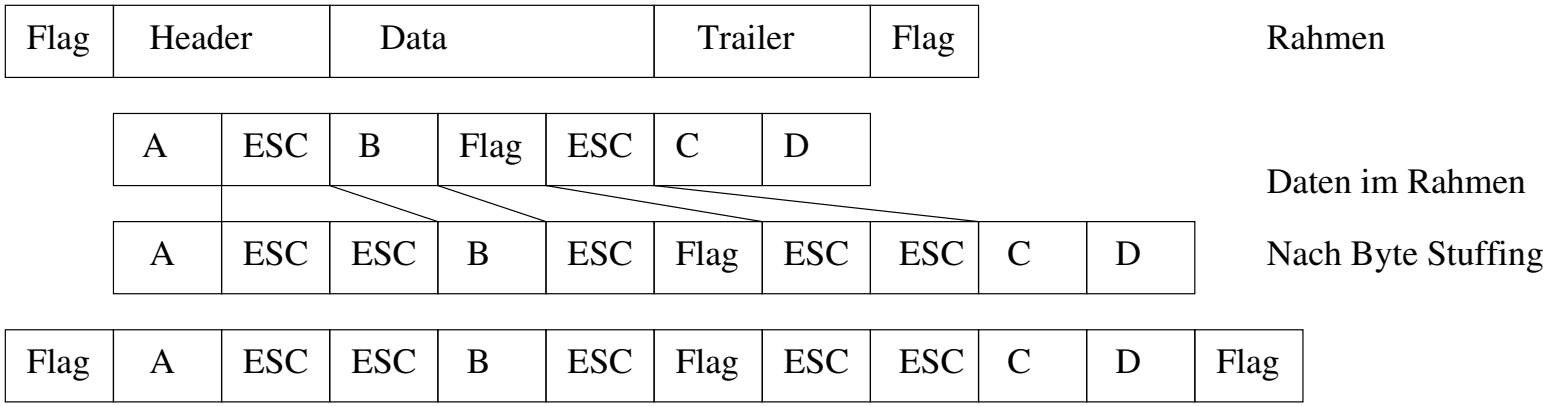
Idee Kennzeichnung von Rahmenanfang und Ende durch spezielle Bitmuster.

Bei zeichenorientierter Bitübertragungsschicht (vgl. z.B. EIA-232) muß die Kennzeichnung durch Zeichen (Flagbytes) erfolgen, da Einzelbits nicht übertragen werden können.

Problem Was passiert, wenn das verwendete Byte im Datenstrom auftaucht?

Byte Stuffing

Ein Byte (nicht das Flagbyte) wird als sogenanntes Escape Zeichen (ESC) gewählt. Dieses Zeichen wird vor der Rahmenbildung vor jedem Flagbyte und vor jedem Escape Zeichen in den Datenstrom eingefügt wird.



Eventuelle Prüfsummen (z.B. im Trailer) werden vor dem Byte Stuffing berechnet und ebenso behandelt.

Start- und Endflag mit Bit Stuffing

- Bei bitorientierter Übertragung kann das Flagbyte Bytegrenzen im Datenstrom überlappen, damit ist keine Erkennung des Rahmenanfangs bei verlorener Synchronisation möglich.
- Daher: Verwende Bitmuster der From **0111...1110** der Länge n als Flag und stelle sicher, daß in den Daten nie $n-2$ mal **1** gesendet wird.
- Erkennt der Sender das $n-3$ mal **1** gesendet wurde, fügt er vor dem nächsten Bit ein Bit
  tt**0** in den Datenstrom ein.
- Analog entfernt der Empfanger die **0**, die auf $n-3$ mal **1** folgt.

Beispiel Bit Stuffing

Flag Bitmuster:	**01110**
Bitfolge Daten:	**0101111101100110**
Daten nach Bit Stuffing:	**0101101101011010011000**
Gesendeter Rahmen	**01110 0101101101011010011000 01110**

Das Muster **01110** taucht nur am Rahmenanfang und Ende auf. Im Rahmen folgen nie mehr als 2 mal **1** aufeinander. Damit kann der Empfänger den Anfang des nächsten Rahmens sicher finden.

Polynome über $\mathbb{Z}_2$

Die Menge der Polynome mit Koeffizienten aus $\mathbb{Z}_2 := \mathbb{Z}/\mathbb{Z}_2$ bilden einen kommutativen Ring.
Beispiele: Seien $p(x) = x^4 + x^3 + 1$ und $q(x) = x^3 + x$, dann ist

$$p(x) + q(x) = x^4 + x + 1$$
$$p(x) - q(x) = x^4 + x + 1$$
$$p(x) * q(x) = x^7 + x^6 + x^5 + x^4 + x^3 + x$$

Bemerkung: Die Addition im $\mathbb{Z}_2$ entspricht der logischen Verknüpfung “exklusiv oder” (XOR), die Multiplikation der Operation “und” (AND).

Beispiel Polynomdivision mit Rest

Division $p(x)/q(x)$ mit Rest ergibt $x+1$ Rest x^2+x+1:

$$
\begin{array}{llllll}
(x^4 & +x^3 & & & +1) & : (x^3+x) = x+1 \text{ Rest } x^2+x+1 \\
x^4 & & +x^2 & & & \\
\hline
 & x^3 & +x^2 & & +1 & \\
 & x^3 & & +x & & \\
\hline
 & & x^2 & +x & +1 &
\end{array}
$$

Bemerkung: Der Grad des Restes ist immer kleiner als der Grad des Divisors.

Anwendung auf Prüfsummen

Sei q ein festes Polynom vom Grad n über $\mathbb{Z}_2$, das sogenannte Generatorpolynom.

Identifiziert man die Bits eines Rahmens mit den Koeffizienten eines Polynoms p über $\mathbb{Z}_2$, dann ist der Rest r der Division $p(x) * x^n / q(x)$ ein Polynom vom Grad $n - 1$. Übertragen werden dann die Koeffizienten von $p(x) * x^n - r(x)$, d.h. die Bits des Rahmens mit angefügten n Prüfbits.

Beispiel

Wir verwenden $n = 3$, $q(x) = x^3 + x$ und wollen **11101** gesichert übertragen.

$$
\begin{array}{cccccccc}
1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\hline
 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
\hline
 & & 0 & 1 & 1 & 0 & 0 & 0 \\
 & & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
 & & & 1 & 1 & 0 & 0 & 0 \\
 & & & 1 & 0 & 1 & 0 & 0 \\
\hline
 & & & & 1 & 1 & 0 & 0 \\
 & & & & 1 & 0 & 1 & 0 \\
\hline
 & & & & & 1 & 1 & 0
\end{array}
\quad : 1010 = 11011 \text{ Rest } 110
$$

Wir übertragen also **11101 110** .

Eigenschaften

Der Empfänger bildet aus den Koeffizienten des empfangenen Rahmens ebenfalls ein Polynom $t(x)$ und berechnet $t(x)/q(x)$. Bei fehlerfreier Übertragung gilt $t(x) = p(x) * x^n - r(x)$.

Sei $e(x) := t(x) - (p(x) * x^n - r(x))$.

1. $p(x) * x^n - r(x)$ wird von $q(x)$ ohne Rest geteilt.
2. Der Rest von $(t(x) - e(x))/q(x)$ ist gleich dem Rest von $e(x)/q(x)$.
3. Ist $x + 1$ ein Faktor von $q(x)$, werden alle Übertragungsfehler mit ungerader Anzahl fehlerhafter Bits erkannt.
4. Mit einem Polynom $q(x)$ vom Grad n und $q(0) = 1$ werden alle Burstfehler bis zu n Bit erkannt.

Beweis zu 3.

Annahme: $e(x)$ hat ungerade Anzahl Terme und $x+1$ teilt $e(x)$.

Bemerkung: Damit der Fehler unerkannt bleibt, muß $e(x)$ von jedem Faktor von $q(x)$ geteilt werden, also insbesondere von $x+1$.

Dann existiert ein Polynom $e'(x)$, so daß $e(x) = (x+1) \cdot e'(x)$ gilt.

Durch Einsetzen erhält man $e(1) = (1+1) \cdot e'(1) = 0$.

Da aber $e(x)$ eine ungerade Zahl an Termen hat, ist $e(1) = 1$, womit die Annahme falsch sein muß.

Beweis zu 4.

Für einen Burstfehler der Länge $k \leq n$ existiert ein $i \geq 0$, so daß gilt:

$e(x) = x^i \cdot (x^{k-1} + \ldots + 1)$.

Da $q(0) = 1$, ist x^i kein Faktor von $q(x)$ für jedes $i > 0$.

Da aber der Grad von $(x^{k-1} + \ldots + 1)$ wegen $k \leq n$ kleiner als der Grad von q ist, wird $(x^{k-1} + \ldots + 1)$ nicht von $q(x)$ geteilt.

Beispiele

Hier sine einige gebräuchliche Generatorpolynome:

- **CRC-16:** Benutzt z.B. von HDLC (ISO 3309)

$$q(x) = x^{16} + x^{15} + x^2 + 1$$

- **CRC-CCITT:** Benutzt z.B. von PPP (RFC 1662)

$$q(x) = x^{16} + x^{12} + x^5 + 1$$

- **CRC-32:** Benutzt z.B. von Ethernet (IEEE 802.3) und PPP

$$\begin{aligned} q(x) = & x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} \\ & + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1 \end{aligned}$$

Bezeichnungen

$W_n = (b_i, i = 0, \ldots, n-1, b_i \in \{0, 1\})$ ein n-stelliges Binärwort.
Dann heißt $\sum_{i=0}^{n-1} b_i$ das Hamming Gewicht von W_n.

Für zwei n-stelliges Binärworte $V_n = (a_0, \ldots, a_{n-1})$ und
$W_n = (b_0, \ldots, b_{n-1})$ heißt das **Hamming Gewicht** von

$$V_n \oplus W_n := (a_0 \oplus b_0, \ldots, a_{n-1} \oplus b_{n-1})$$

der **Hamming Abstand** von V_n und W_n. Hierbei bezeichnet $\oplus$ die XOR-Verknüpfung.

Das Hamming Gewicht von $(1, 0, 1, 1, 0)$ ist 3.
Der Hamming Abstand von $(1, 0, 1, 1, 1)$ und $(1, 1, 0, 1, 1)$ ist 2.

Bestehe das Quellalphabet auf 2^n n-stelligen Binärworten, die so auf $n + r$-stellige Binärworte abgebildet werden sollen, daß k falsch übertragene Bit korrigiert werden können.

Sei z.B. $n = 1$, $k = 1$. Der Versuch mit $r = 1$ führt z.B. zu

$$(0) \rightarrow (0, 0)$$
$$(1) \rightarrow (1, 1)$$

Wird hier ein Bit falsch übertragen, kann nicht mehr entschieden werden, was das Ursprungswort war.

Der Versuch mit $r = 2$ führt dagegen z.B. zu

$$(0) \rightarrow (0, 0, 0)$$
$$(1) \rightarrow (1, 1, 1)$$

Hier ist eine Dekodierung auch bei einem falschen Bit möglich.

Allgemein gilt: Seien V_n und W_n zwei n-stellige Zeichen des Quellalphabetes.

Seien V_{n+r}, W_{n+r} die zugehörigen $n+r$-stelligen Zeichen des Kodes.

Damit bei Fehlübertragung von k Bit V_{n+r} und W_{n+r} wieder sicher den Zeichen des Quellalphabetes zugeordnet werden können, muß der Hamming Abstand von V_{n+r} und W_{n+r} mindestens $2 \cdot k + 1$ betragen.
Damit eine solche Abbildung existiert muß gelten (Hamming Bedingung, 1950):

$$2^n \cdot \sum_{i=0}^{k} \binom{n+r}{i} \leq 2^{n+r}$$

Korrektur von Einzelfehlern

Für $k = 1$ folgt aus der Hamming Bedingung
$2^n \cdot (1 + n + r) \leq 2^{n+r}$, also $(1 + n + r) \leq 2^r$.

Beispiel: Für $n = 4$ werden wegen

$$1 + 4 + 3 \leq 2^3$$
$$\text{aber } 1 + 4 + 2 \not\leq 2^2$$

mindestens $r = 3$ Prüfbits gebraucht.

Für Einzelfehler hat Hamming ein Kodierverfahren angegeben, daß diese Schranke einhält.

Hamming Code

Betrachte ein Schema mit $n + r$ Spalten $1, \ldots, n + r$. Bezeichne weiter $\odot$ die bitweise AND Verknüpfung. Seien die r Prüfbits in den Spalten $2^0, 2^1, \ldots, 2^{r-1}$ und die n Bit Nutzdaten in den übrigen Spalten, dann bildet Spalte 2^i die Parität für alle Spalten s mit $s \odot 2^i = 2^i$.

Beispiel: Prüfbit 0 in Spalte $2^0 = 1$ bildet die Parität für die Spalten $1, 3, 5, 7, \ldots$.

Prüfbit 1 in Spalte $2^1 = 2$ bildet die Parität für die Spalten $2, 3, 6, 7, 10, 11, \ldots$.

Umgekehrt wird Spalte $25 = 2^4 + 2^3 + 2^0$ abgesichert durch die Prüfbits in den Spalten $1, 8$ und 16.

Beispiel

Seien $n = 4$ und $r = 3$ und wir benutzen gerade Parität.

Spalte(dezimal)	1	2	3	4	5	6	7
Spalte(binär)	001	010	011	100	101	110	111
Prüfbit	↓	↓		↓			
1011	0	1	1	0	0	1	1
0110	1	1	0	0	1	1	0
0001	1	1	0	1	0	0	1

Bemerkung: Bis auf die Prüfbits sind alle Bits in mindestens zwei Paritätsbits eingeflossen.

Dekodieren der Nachricht

- Der Empfänger berechnet analog zum Sender die Prüfbits.
- Sind alle Prüfbits identisch mit den empfangenen Bits, ist die Nachricht fehlerfrei übertragen worden.
- Im Fall, daß das Bit in Spalte $s = \sum_{i=0}^{r-1} b_i 2^i$ falsch übertragen worden ist, stimmen die empfangenen Paritätsbits in den Spalten $\{b_i 2^i,\ b_i = 1, i = 0, \ldots, r-1\}$ nicht mit den gesendeten Daten überein.
- Der Empfänger invertiert das Bit in Spalte s und extrahiert die Nutzdaten.

Motivation

- Sendet der Sender schneller als der Empfänger die Daten verarbeiten kann, gehen Teile verloren.
- Kann ein Rahmen nicht dekodiert werden, fehlt ebenso ein Stück im Datenstrom.
- Höhere Schichten (Transportschicht) können Fehler auf unteren Schichten bei Bedarf korrigieren.
- Die Sicherungsschicht kann durch Flußkontrolle den Prozeß auf Ebene der Rahmen optimieren. Dadurch müssen nicht komplette Pakete wiederholt übertragen werden.
- Die Flußkontrolle der Sicherungsschicht sollte abschaltbar sein, da für einige Dienste (z.B. Sprachübertragung) Verzögerungen schlimmer sind als Datenverluste.

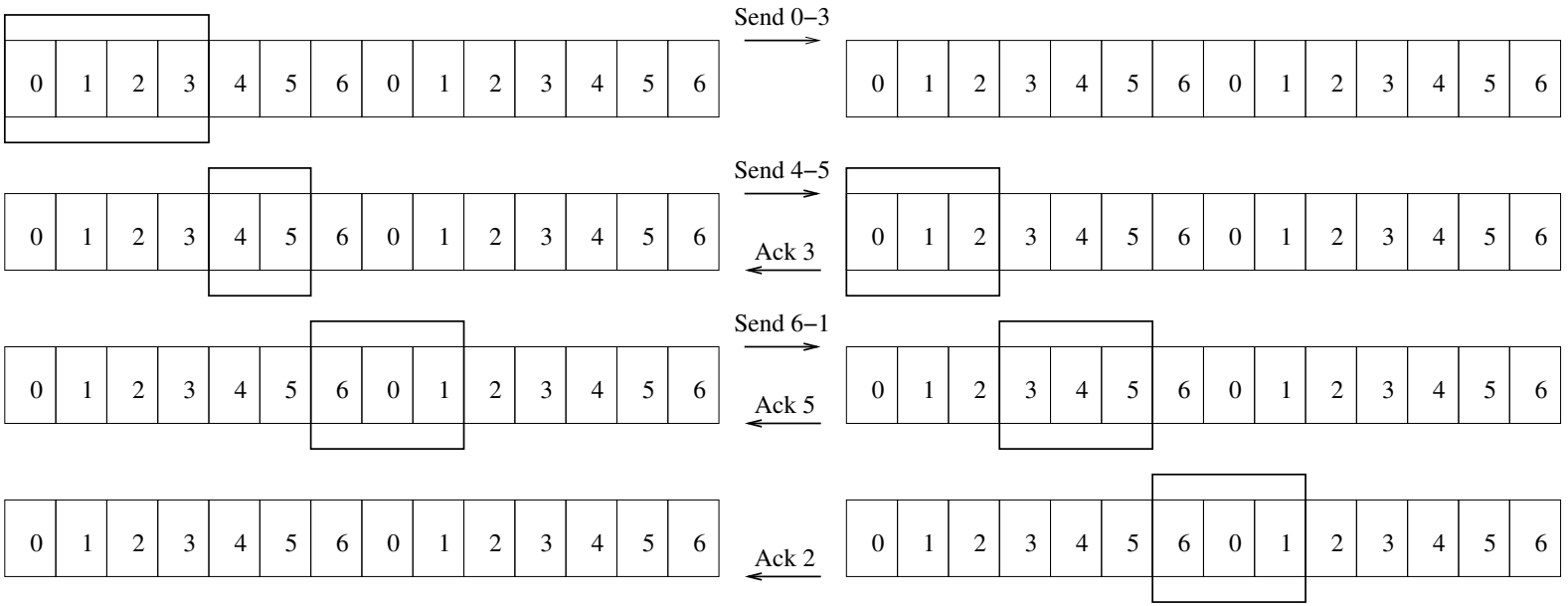
Stop-and-Wait Protokoll

- Nach jedem übertragenen Rahmen wartet der Sender auf eine Bestätigung (ACK) des Empfängers.
- Vorteil: Extrem einfach zu implementieren, benötigt beim Empfänger wenig Resourcen.
- Nachteil: Das Protokoll ist ineffizient.

Sliding Window Protokoll

- Der Empfänger hat einen Puffer (sog. Fenster) mit Platz für $n-1$ Rahmen.
- Rahmen werden mit $0, 1, \ldots, n-1, 0, 1, \ldots, n-1, 0, \ldots$ modulo n numeriert.
- Der Sender kennt den Wert n und weiß, daß er $n-1$ Rahmen senden darf, ohne daß einer der Rahmen quittiert worden ist.
- Der Empfänger schickt im ACK die Nummer des ersten nicht verarbeiteten Rahmens.

Beispiel mit Puffer für 6 Rahmen,d.h. $n = 7$:



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

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

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

Die folgenden Fälle sind relevant:

- Fehlerhaft übertragener Datenrahmen

  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 (2)

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

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($k+l+1$).

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)
- Random Access
- Tokenverfahren (z.B. Token Ring, FDDI)

Alohanet (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.

Durchsatz des Alohanetzes

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 Alohanetzes (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.

Ethernet

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.

Einführung

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} gesendet.

Address Immer **11111111**

Control Immer **00000011**

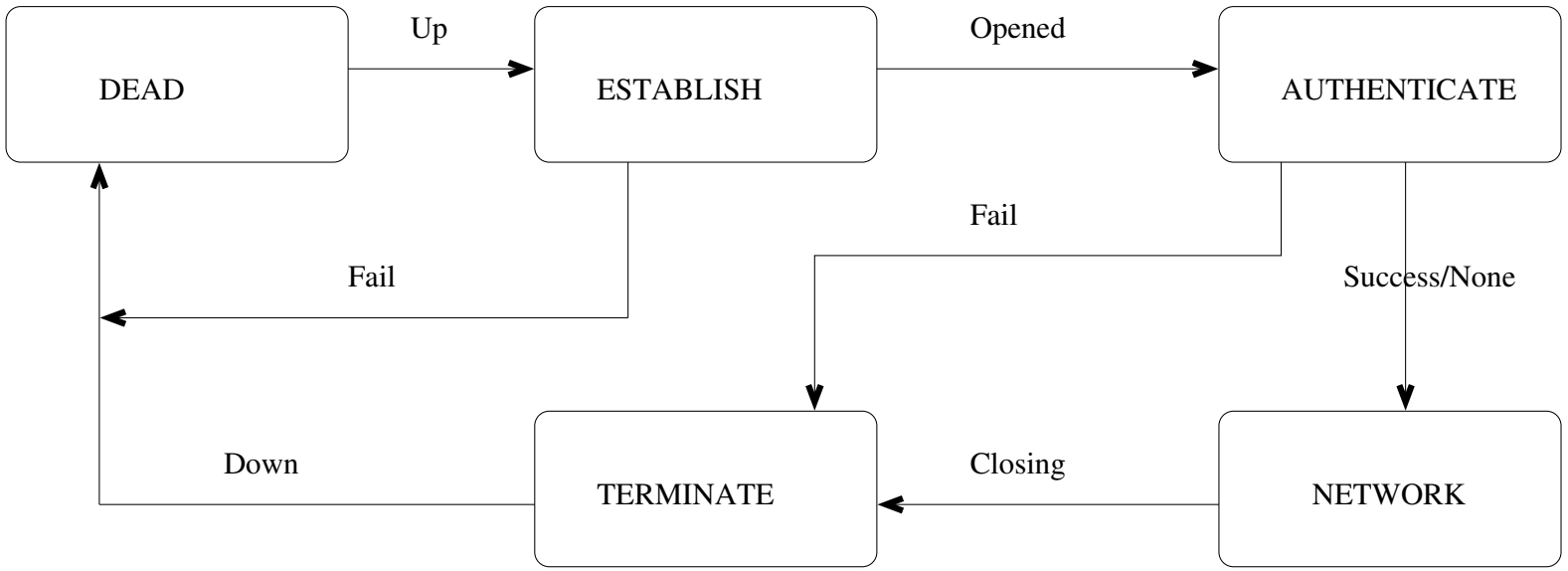
Protocol 16Bit Protokollspezifikation gemäß RFC1700, PPP DLL Protocol Numbers

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 Packete 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 Packete dienen dazu, die Verbindung zu schließen.

Einführung

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

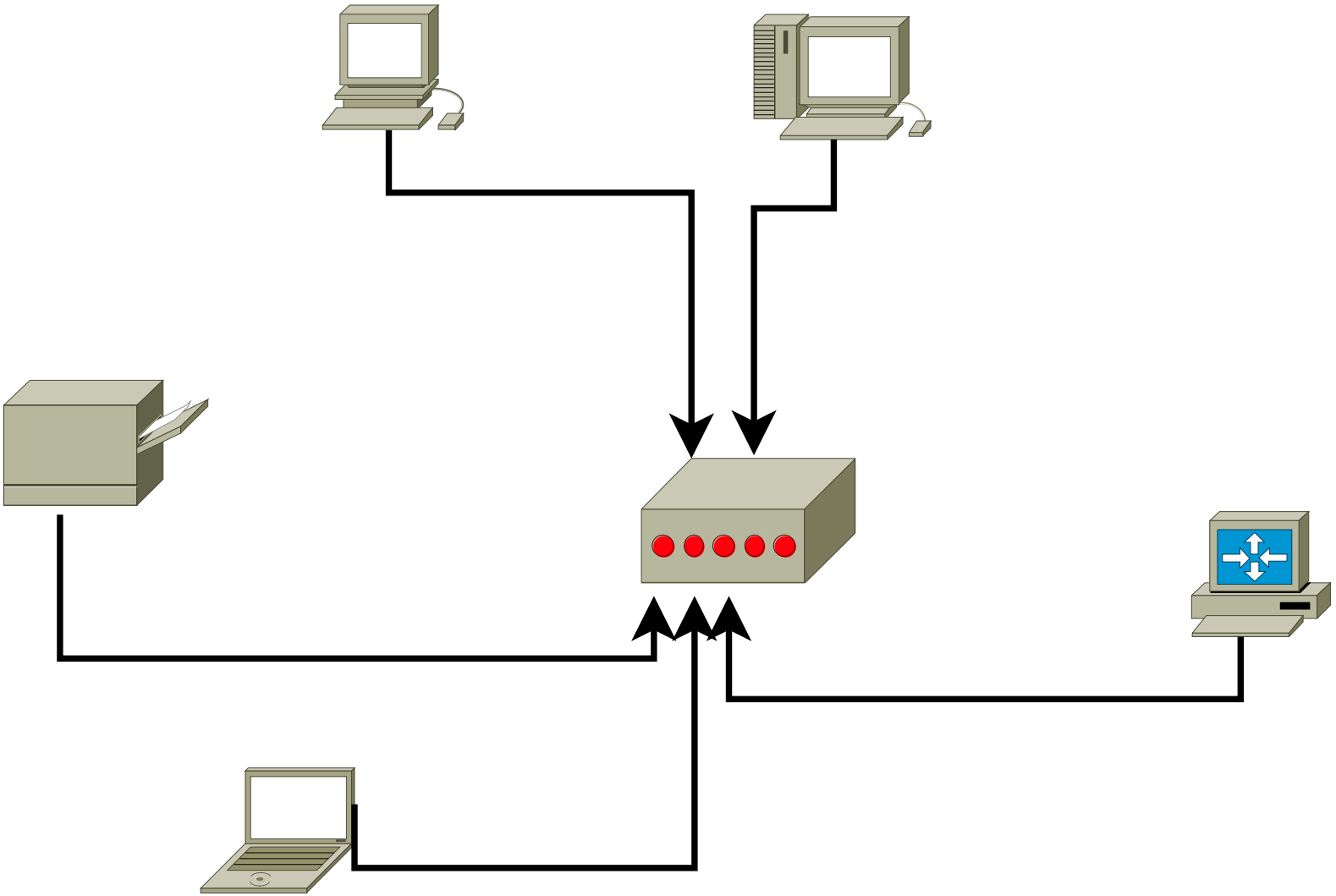
- **Preamble/SOF:** 7 Byte zur Synchronisation **10101010**, dann Start Of Frame Byte.
- **Destination/Source:** Adressen von Zieladapter und Quelladapter, jeweils 6 Byte
- **Length/Type:** 2 Byte Typ der gekapselten Daten benutzt. Werte gemäß RFC1700, Ether Types. Der IEEE Standard erlaubt hier die 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. $03 : 00 : 00 : 20 : 00 : 00$ 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)
- Packete sind nur auf den Links sichtbar, die sie benötigen.

Weiterleitung von Packeten

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