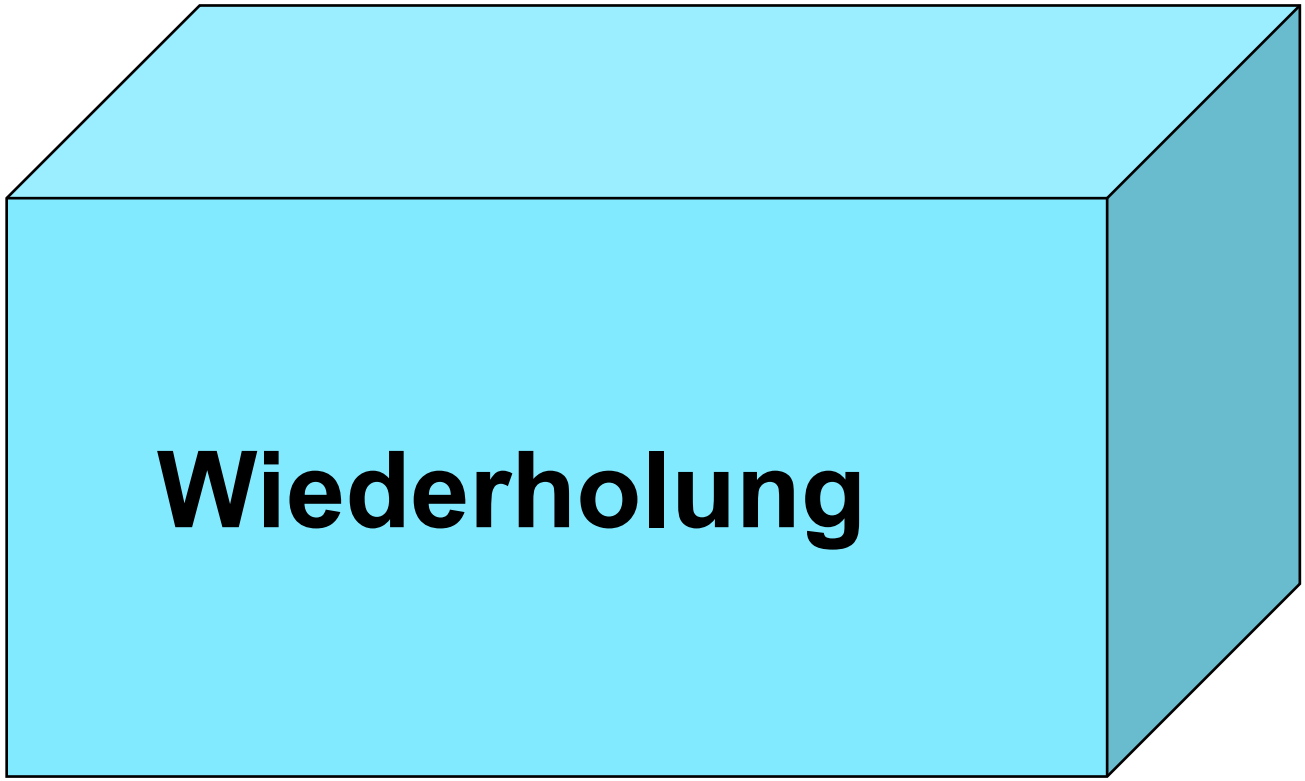
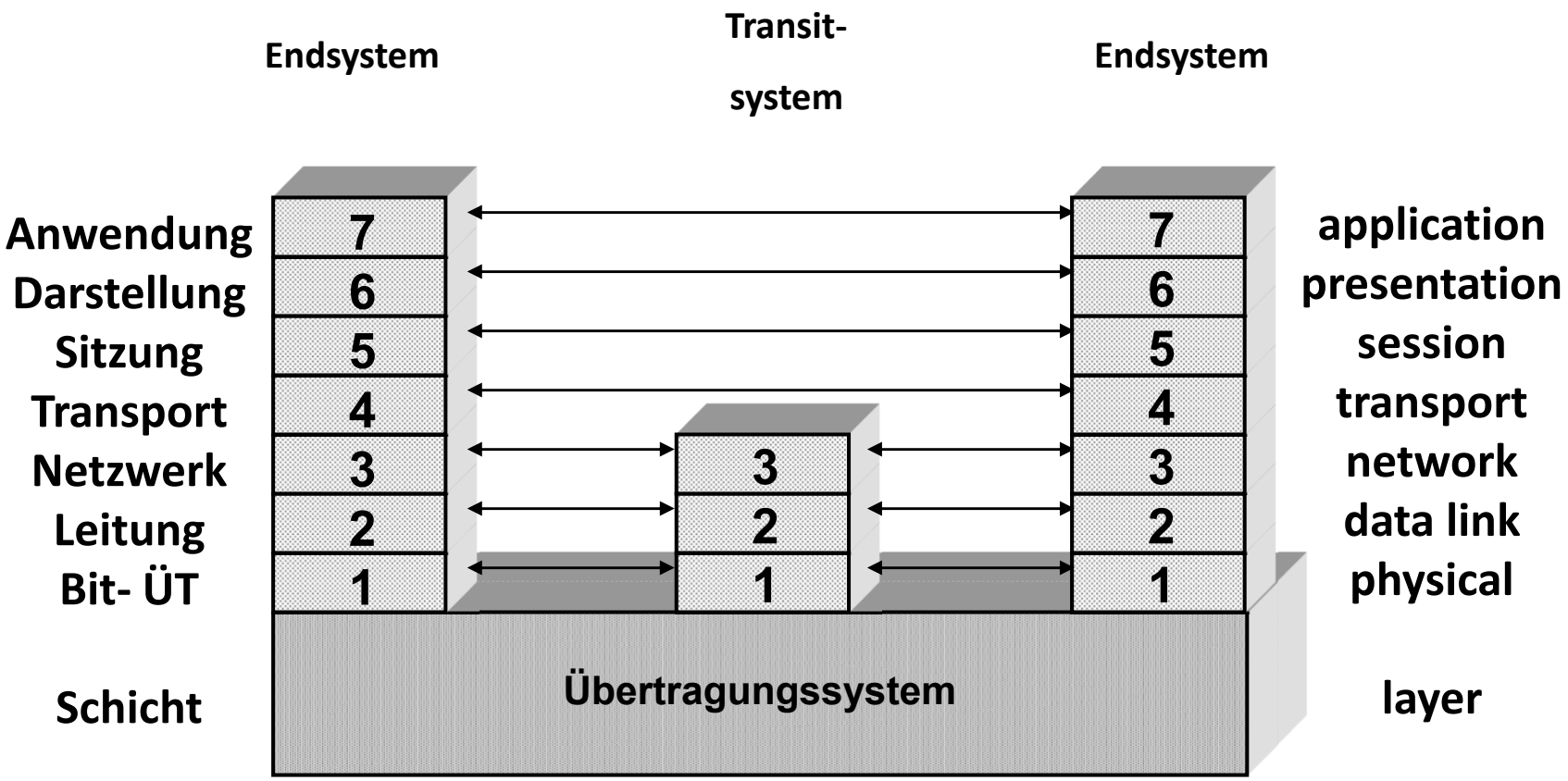

Wiederholung

## Mögliche Aufgaben in der Klausur:

- Rechenaufgaben:
  - z.B. Aufgaben zu den Kabelparametern, eine CRC Berechnung (ähnlich Übungsblatt 3, Aufgabe 4), Aufgaben zur Bildung von Subnetzen bei IPv4 oder IPv6 o.ä.
- kleinere Verständnisaufgaben/ einige Multiple Choice Fragen:
  - z.B. zum OSI-Modell, zu den Eigenschaften eines Switch, zu CSMA/ CA, UDP, TCP, DNS, symmetrisch/ asymmetrische Verschlüsselung, SSH, Paket-/ Leitungsvermittlung ...
- Aufgaben zum Praktikum/ zu den zusätzlichen Übungsblättern
  - z.B. ein Szenario ähnlich zu den Szenarien der Praktikumsblätter/ der zusätzlichen Übungsblätter 9 und 10, wie läuft die Adressauflösung ab, wie sehen die Neighbor Caches und/ oder Switching-Tabellen nach einer Kommunikation aus, geeignete IP Adressen von Stationen in einem Netz, Gültigkeitsbereich einer link-lokalen IPv6 Adresse …

# Übersicht: OSI Referenzmodell



# Kabelparameter: Dämpfung

$$a = 10 \log_{10}(P_{in} / P_{out})$$

a Dämpfung (attenuation)

$P_{in}$ Leistung des Eingangssignals

$P_{out}$ Leistung des Ausgangssignals

übliche Einheit: dB/100m

# Kanalkapazität (ohne Einbezug von Störungen)

**Kanalkapazität bei 2 Signalebenen bzw. Signalwerten:**

$C = 2\,B$ *bit*

*B* *Bandbreite*
*C* *Kanalkapazität*

**Kanalkapazität bei M Signalebenen (bei „besonderer“ Kodierung)**

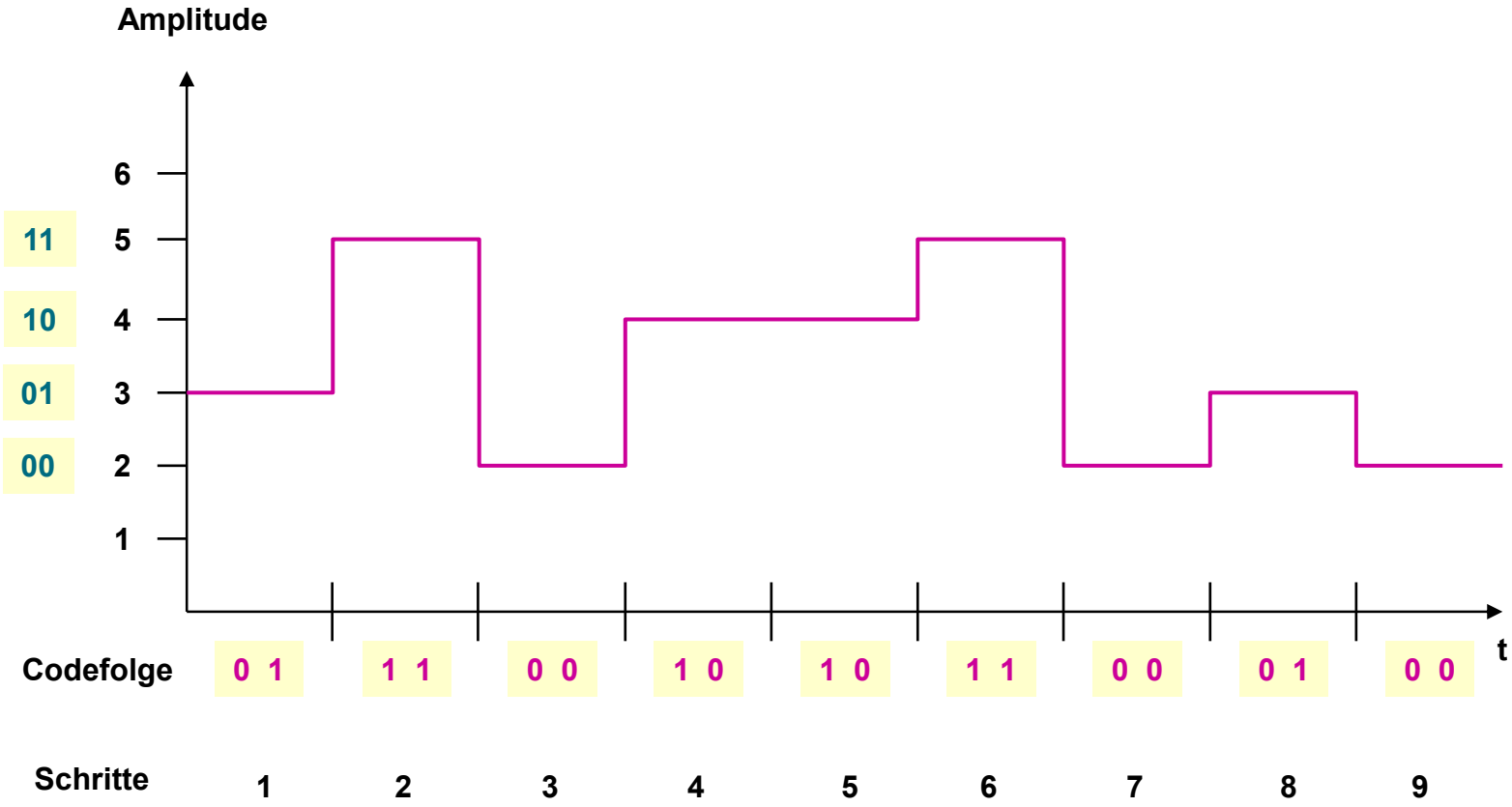
$C = 2\,B \log_2(M)$ *bit*

*B* *Bandbreite*
*C* *Kanalkapazität*

**Baudrate**
**= Schrittgeschwindigkeit**
**= Zahl der Signalparameter-Zustandswechsel**

**Bei 2 Signalwerten ("0" = 0 Volt, "1"= 5 Volt): Bitrate = Baudrate**

# Beispiel: Digitales Signal mit 4 Signalwerten



# Kanalkapazität (unter Einbeziehung von Störungen )

**Rauschabstand:**

$SNR = 10 \log_{10} (P_S/P_n)$

- SNR signal to noise ratio
- $P_S$ Leistung des Nutzsignals
- $P_n$ Leistung des Rauschsignals (noise)

**Gesetz von C. Shannon (1948)**

$C = B \log_2 (1 + P_S/P_n)$

- C Kanalkapazität
- B Bandbreite
- $P_S$ Leistung des Nutzsignals
- $P_n$ Leistung des Rauschsignals (noise)

# Aufgaben mit den Kabelparametern

**1.) Dämpfung**

**a** **Dämpfung (in dB)**

**$P_{in}$** **Eingangsleistung (oder $U_{in}$ Eingangsspannung)**

**$P_{out}$** **Ausgangsleistung (oder $U_{out}$ Ausgangsspannung)**

**2 von 3 Werten sind gegeben und ev. Umrechnung der Dämpfung auf eine Länge**

**2.) SNR ebenso**

**3.) Kanalkapazität**

**Kanalkapazität C, Bandbreite B, Signalebenen M**

**2 Werte sind gegeben und der dritte muss berechnet werden**

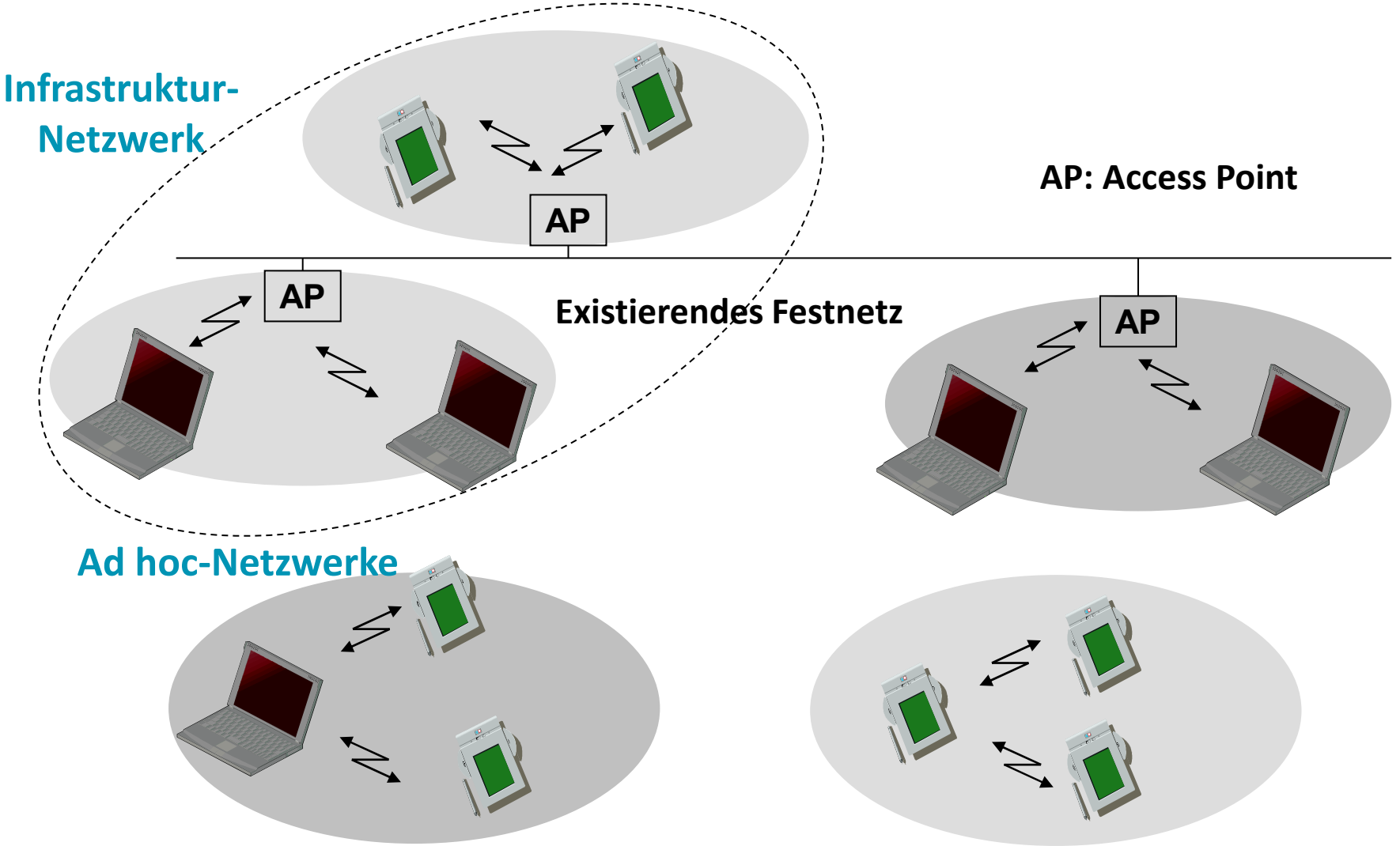
**Oder nach Shannon, dann sind die Kanalkapazität, die Bandbreite, Leistung des Nutzsignals $P_S$ und Leistung des Rauschsignals (noise) $P_n$ wichtig**

***Eher nicht:***

***4.) Signallaufzeit***

***Länge oder Delay ist gegeben und der andere Wert ist zu berechnen***

# Grundlegende Architektur von WLAN: Infrastruktur- und Ad hoc-Netzwerke



# Carrier Sense Multiple Access with Collision Avoidance (CSMA/ CA)

- Eine normale Antenne kann entweder Daten senden oder Daten empfangen.
- Daher wird in WLAN ein abgewandeltes Zugriffsverfahren verwendet: CSMA/ CA (Carrrier Sense Multiple Access with Collision Avoidance)
- Jede Datenübertragung wird bei CSMA/ CA vom Empfänger mit einem Acknowledge (ACK) bestätigt (Ausnahme: Broadcast Nachrichten).
- Damit es beim Senden des ACK nicht zu Kollisionen kommen kann, wird das ACK nach einer sehr kurzen Wartezeit gesendet, die kürzer ist die minimale Wartezeit, die bei einer Datenübertragung auftreten kann.
- Wartezeit vor einer normalen Daten Übertragung: DIFS: Data-Inter-Frame Spacing plus zufällige Wartezeit
- Wartezeit vor der Sendung von ACK (und einigen anderen Steuernach-richten) SIFS: Short Inter-Frame Spacing

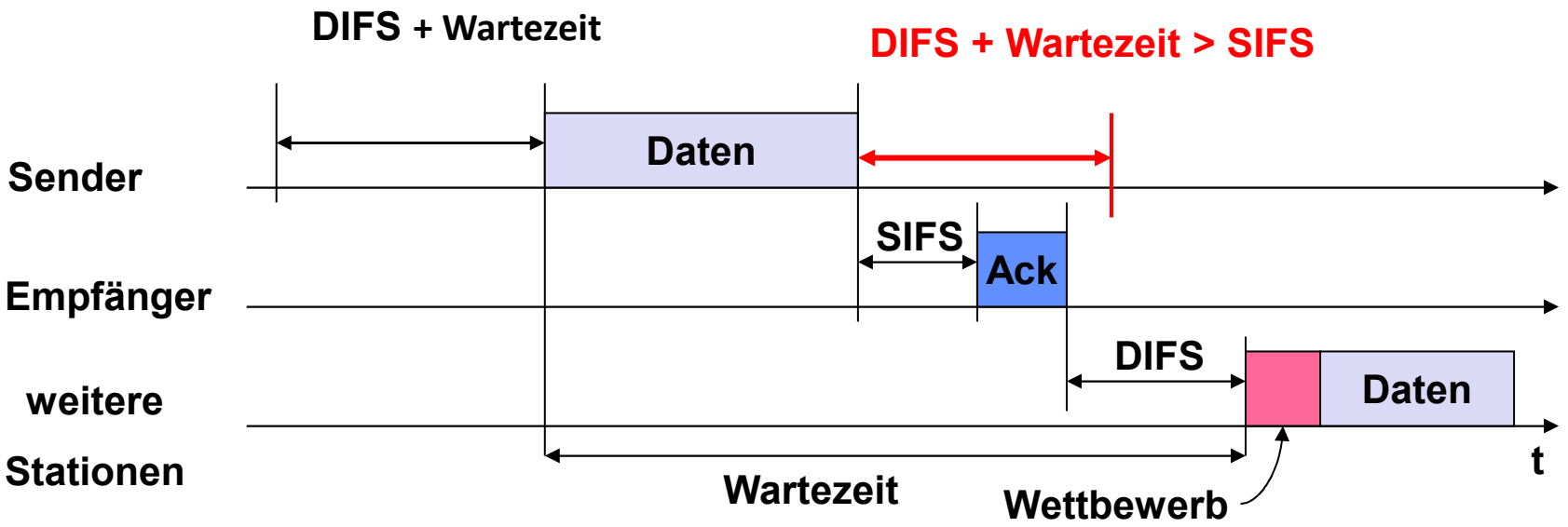
Es gilt: DIFS > SIFS

# Der grundsätzliche Ablauf von CSMA/ CA

- Die sendewillige Station hört das Medium ab.
- Ist das Medium frei, darf sie nach DIFS plus zufälliger Wartezeit mit der Sendung ihrer Daten beginnen.
- Ist das Medium nicht frei, wartet sie, bis das Medium frei wird und sendet dann nach DIFS plus einer zufälligen Wartezeit.
- Der Empfänger quittiert die Sendung mit einem ACK. Nach Erhalt des ACK ist die Datenübertragung abgeschlossen. (Das „Wettbewerbsfenster“ wird auf den minimalen Bereich gestellt.)
- Bei Ausbleiben des ACK erneutes Senden der Daten mit neuem (nicht bevorzugten!) Medienzugriff mit verdoppeltem Wettbewerbsfenster.
- Beginnt eine andere Station vorher zu senden, muss die sendewillige Station die Sendung ihrer Daten verschieben.
- Allerdings kann sie beim nächsten Sendeversuch ihre „Rest“-Wartezeit verwenden.

=> Verbesserung der Fairness!

# Ablauf einer Punkt-zu-Punkt Datenübertragung

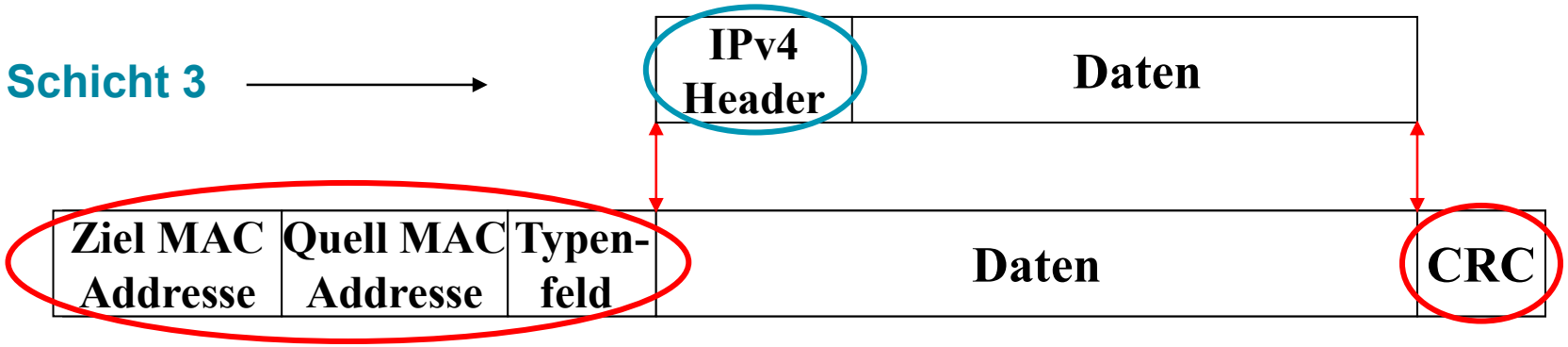



**Medienzugriff wie beschrieben mit Bestätigung jeder Übertragung (nur bei Broadcast Nachrichten keine Bestätigung!), Feststellen einer Kollision durch Ausbleiben des ACK.**

# Einordnung IP

- IP ist ein verbindungsloser Nachrichtentransportdienst
  (ohne Fehlerkorrektur, ohne Empfangsbestätigung, ohne Sicherung der Reihenfolge der übertragenen Datenpakete und ohne eine „spezielle Route“ festzulegen)
- Das IP Protokoll ist OSI Schicht 3 zuzuordnen, es definiert die IP Adressen und den Aufbau von IP Datenpaketen (Datagrammen)

**Auf Schicht 3: Ende-zu-Ende Adressierung**
**d.h. der tatsächliche Empfänger**




**Auf Schicht 2 wird der nächste „Hop“ adressiert**

# Darstellung von IPv4 Adressen

- **Adresslänge bei IPv4: 4 Byte bzw. 32 Bit**
- **Darstellung der 32 Bit IP-Adresse durch 4 durch Punkte getrennte Dezimalzahlen (eine für jedes Byte)**

  **z.B. 01100101 . 00011110 . 00000110 . 00010100**

  **= 101.30.6.20**

# Anwendungsbeispiel: Subnetzbildung

- Gegeben ist die Netzadresse 192.32.8.0. Es werden 6 Subnetze benötigt, wobei das größte Netz 30 Rechner enthält

⇨ Zur Adressierung von 6 Netzen werden 3 Bit des Rechneradressraums für eine Erweiterung der Netzadresse verwendet

⇨ Zur Adressierung von 30 Rechnern werden 5 Bit benötigt

- Die Subnetzmaske ergibt sich damit zu: 255.255.255.224

  11111111 . 11111111 . 11111111. 111 00000

- Um was für eine Adresse handelt es sich bei 196.32.8.130?
- Wie lauten die Subnetzadressen der Abteilungen?
- Die Abteilungen erhalten die folgenden Adressen:

| | | |
|---|---|---|
| 11000100.00100000.00001000. 001 00000 | = | 196.32.8.32 |
| 11000100.00100000.00001000. 010 00000 | = | 196.32.8.64 |
| 11000100.00100000.00001000. 011 00000 | = | 196.32.8.96 |
| 11000100.00100000.00001000. 100 00000 | = | 196.32.8.128 |
| 11000100.00100000.00001000. 101 00000 | = | 196.32.8.160 |

# Anwendungsbeispiel (2)

- **Alle Subnetze haben die gleiche Subnetzmaske**
- **Jedes Subnetz hat eine eigene Subnetzadresse**

  **UND eine eigene Broadcastadresse**
- **Pro Abteilung bleiben 5 Bits für die Adressierung der Rechner => $2^5$ = 32 Zustände**

  **Aber: der Zustand 00000 ist bereits für die Netzadresse verwendet und der Zustand 11111 für die Broadcast- Adresse**

  **=> es können $2^5$ – 2 = 30 Rechner pro Abteilung adressiert werden.**

**Welche beiden Randbedingungen sind grundsätzlich bei einer Subnetzbildung zu beachten?**

**Die Anzahl der benötigen Subnetze UND die Anzahl der Rechner im größten Subnetz!**

# Anwendungsbeispiel (3)

| | | | |
|---|---|---|---|
| • | IP-Adresse der 1. Abteilung: | 11000100.00100000.00001000. 001 00000 | = 196.32.8.32 |
| | mit der Broadcast-Adresse: | 11000100.00100000.00001000. 001 11111 | = 196.32.8.63 |
| • | IP-Adresse der 2. Abteilung: | 11000100.00100000.00001000. 010 00000 | = 196.32.8.64 |
| | mit der Broadcast-Adresse: | 11000100.00100000.00001000. 010 11111 | = 196.32.8.95 |
| • | IP-Adresse der 3. Abteilung: | 11000100.00100000.00001000. 011 00000 | = 196.32.8.96 |
| | mit der Broadcast-Adresse: | 11000100.00100000.00001000. 011 11111 | = 196.32.8.127 |
| • | IP-Adresse der 4. Abteilung: | 11000100.00100000.00001000. 100 00000 | = 196.32.8.128 |
| | mit der Broadcast-Adresse: | 11000100.00100000.00001000. 100 11111 | = 196.32.8.159 |
| • | IP-Adresse der 5. Abteilung: | 11000100.00100000.00001000. 101 00000 | = 196.32.8.160 |
| | mit der Broadcast-Adresse: | 11000100.00100000.00001000. 101 11111 | = 196.32.8.191 |

## IPv6 - Adressen

- Länge der Adressen: 128 Bit
- Darstellung in 8 Gruppen von 4 x 4 Bit davon je 4 Bit in Hexadezimaldarstellung getrennt durch ":"
- Beispiel: 2001:0005:0000:0000:0000:0000:EEC1:0008
  (entsprechend: 0010 0000 0000 0001: ... )
- Eine Folge von Nullen kann weggelassen werden und wird dann durch :: gekennzeichnet

  Bei dem Beispiel von oben ergibt sich damit: 2001: 0005 :: EEC1:0008
- Nullen am Anfang einer Hexadezimalgruppe d.h. „führende" Nullen können ebenfalls weggelassen werden:

  Für das Beispiel von oben folgt damit: 2001: 0005::EEC1: 0008
  = 2001: 5:: EEC1: 8
- Die IPv6-Adressen bezeichnen (wie bei IPv4) einzelne Interfaces

## IPv6: Globale Unicast-Adressen

- Struktur einer globalen Unicast-Adresse

| Globales Routingpräfix (= Routing Bereich) | Subnetz-ID | Schnittstellen-ID |
|---|---|---|
| 48 (… 64) Bit | 16 (… 0) Bit | 64 Bit |

- Alle globalen Unicast Adressen, die nicht mit 000 beginnen, haben eine Schnittstellen-ID mit einer Länge von 64 Bit
- Viele global gültige Adressen, die zur Zeit vergeben werden, beginnen mit 2001
- Wie kann man bei gegebenem Routingpräfix bei einer IPv6 Adresse eine Subnetzbildung durchführen …??

## Lokale Unicast-Adressen (1)

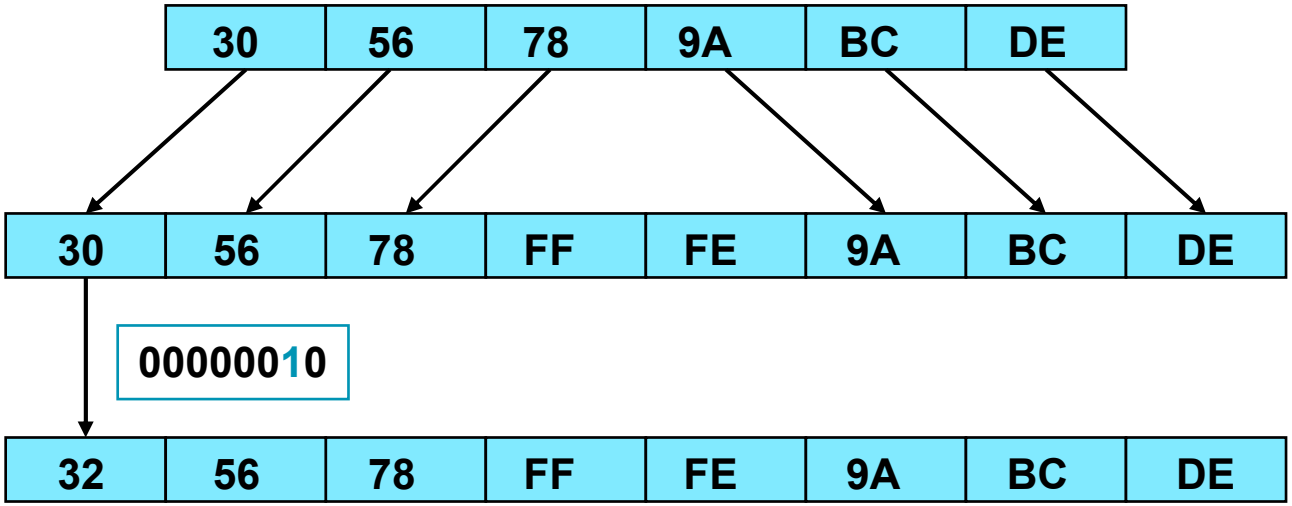

## Verbindungslokale Adressen (Link Local)

- Beginnen mit FE80 und werden immer automatisch generiert und z.B. für Zwecke der Nachbarerkennung verwendet
- Gültigkeitsbereich ist die lokale Verbindung
- Entsprechen den automatisch auf Computern konfigurierten IPv4 Adressen

## Beispiel zur Bildung eines Interface-ID:

- Generierung aus der MAC Adresse nach EUI-64

48 Bit → 64 Bit




Das vorletzte Bit im ersten Byte zeigt die „globale Eindeutigkeit“ einer MAC Adresse an. Es wird mit „OR“ 00000010 verknüpft und auf „lokale Gültigkeit“ gesetzt.

## Beispiel: Bildung einer lokal gültigen Adresse

- **Z.B. mit Präfix für „Lokal Link“ Adresse: FE 80 ( = 1111 1110 10 ),**

  **dann 54 Bit 00 .. 00 und**

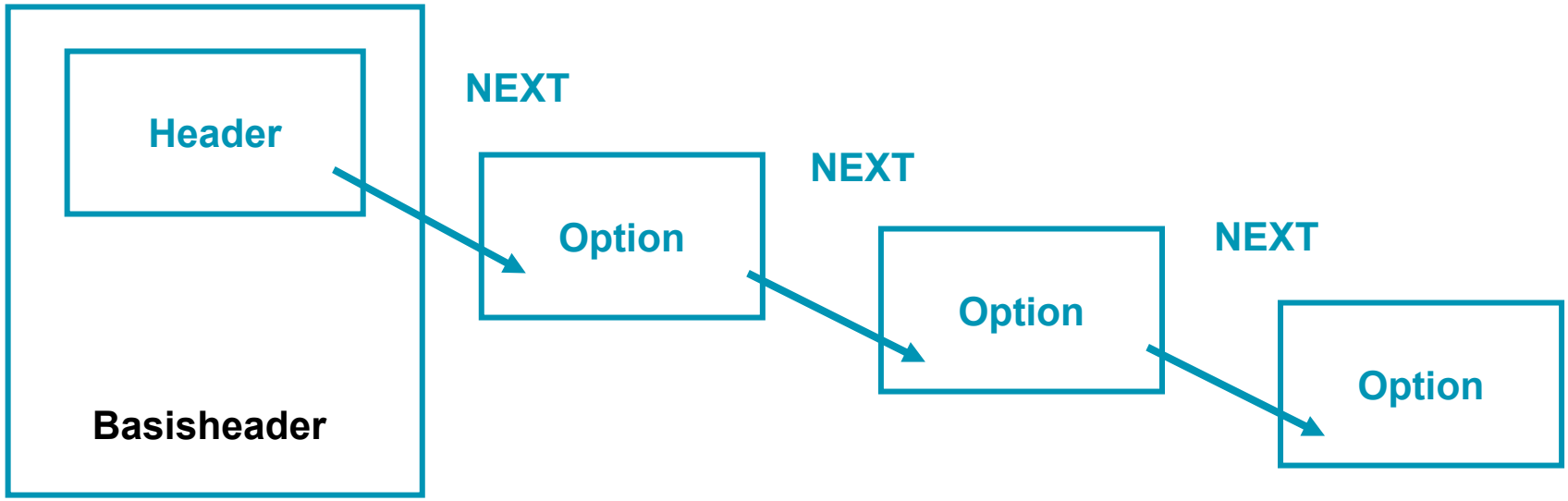
  **64 Bit eindeutige Schnittstellen ID aus MAC-Adresse nach EUI-64**

| 1111 1110 10 | 000 ... 000 | Eindeutige Schnittstellen-ID |
|---|---|---|
| 10 Bit | 54 Bit | 64 Bit |

## Prinzipieller Aufbau des IPv6 Headers

- Es gibt einen Basisheader,der nur die wichtigsten und für den Transport notwendigen Daten enthält
- Die Optionen befinden sich in hierarchisch verketteten Headern, die nicht von allen Netzkomponenten ausgewertet werden, sondern nur dort, wo die Information auch benötigt werden!!!

  ⇒ Die Reihenfolge, in der die Header auftreten können, ist festgelegt!



# Grundfunktionen und -merkmale eines Switch

**Ein Switch ist eine Netzkomponente, der auf Schicht 2 arbeitet.**

**Switching-Tabelle enthält MAC-Adressen und Portnummern**

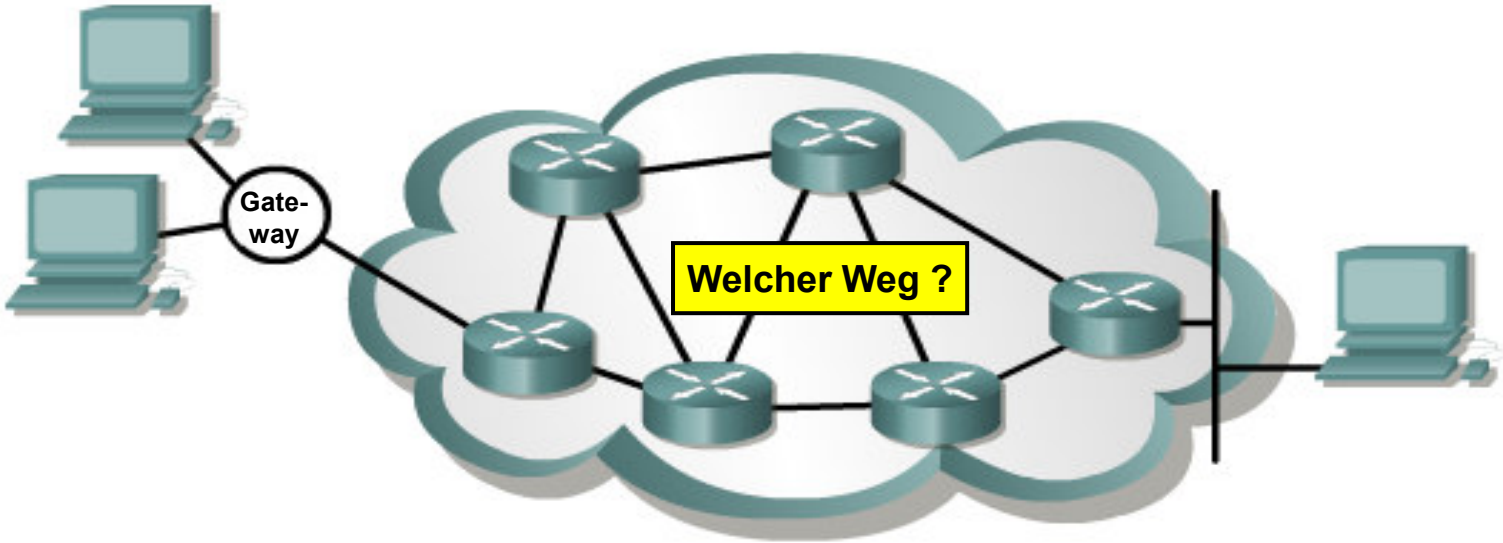
**Grundfunktionen und -merkmale eines Switch:**

- **Lasttrennung durch Frame-Filterung,**
- **Auftrennung von Kollisionsdomänen,**
- **Transparenz des Switch für die angeschlossenen Stationen,**
- **Weiterleitung von Broadcast- und Multicast-Nachrichten (führt zum Begriff der Broadcast-Domäne).**
  **Eine Broadcastdomäne wird typischerweise durch einen Router begrenzt, da Router keine (Layer2)-Broadcast-Nachrichten weiterleiten,**
- **Selbstlernend, selbstkonfigurierend (auf der Basis der MAC Source-Adressen) und Agingmechanismus**

**sowie:**

- **Unterstützung redundanter Netzwerkpfade (→ Spanning Tree),**
- **Bridge nutzt Store-and-Forward Mechanismus in Abgrenzung zum Switch, Switch ist auf Geschwindigkeit optimiert, ansonsten gleiche Funktionalität.**

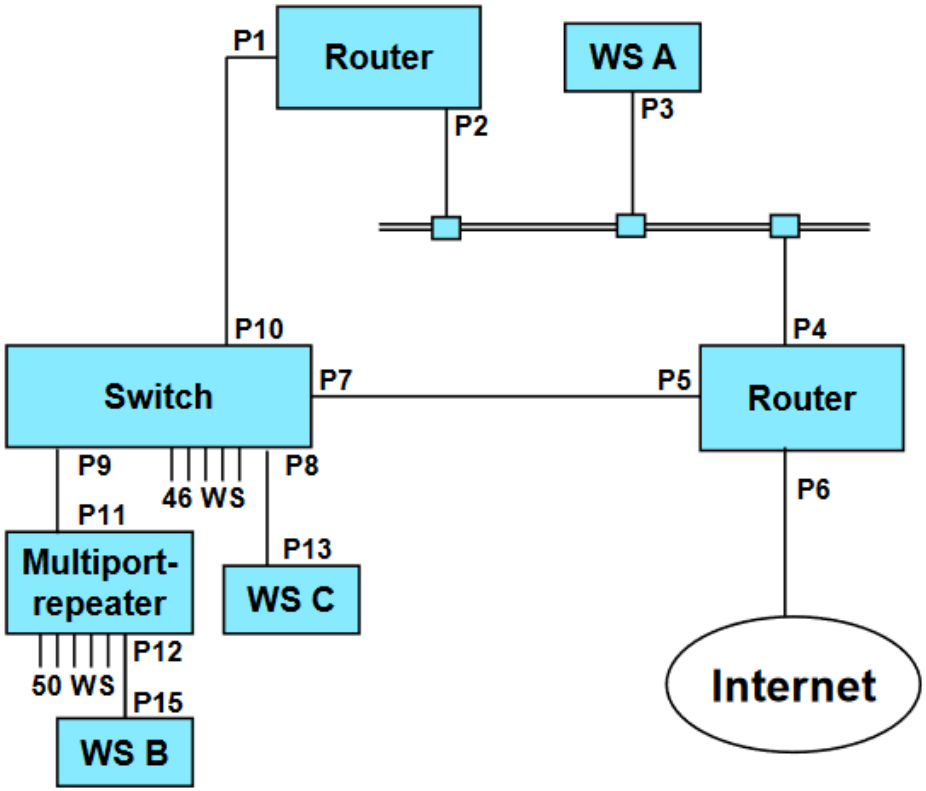
# Der Router als klassische Schicht 3 – Komponente (1)




Quelle: Schulungsunterlagen Cisco Academy

- Ein Router arbeitet auf Schicht 3 und wertet die IP-Protokoll-informationen aus
- Seine wichtigste Aufgabe ist die Wegewahl innerhalb des Netzes
- Routing-Tabelle enthält Zuordnungen von IP-Adresse/ Adressbereichen zu Ports

# Kollisions- und Broadcastdomänen




Aufgabe

Gegeben sei das links beschriebene Netz:

(a) Welche Ports gehören zur selben Kollisionsdomäne?

(b) Welche Ports gehören zur selben Broadcastdomäne?

# Statisches Routing

- Die einfachste Möglichkeit für den Aufbau einer Routing-Tabelle:

⇒ **statisches Routing:**

Die Einträge der Routing-Tabelle werden manuell eingeben, d.h. das Routing basiert auf fest vorgegebenen Informationen

- Einsatzgebiet: in kleineren Netzen mit wenigen Routern
- Änderungen müssen von Hand eingegeben werden d.h. die Tabellen müssen „gepflegt“ werden
- Bestimmte Adressen werden immer an bestimmte, festgelegte Interfaces geleitet
- Unbekannte Adressen werden an ein „Standard-Gateway“ geleitet, das in der Regel den Zugang zum Internet bildet

# Dynamisches Routing

- **Zweite Möglichkeit für den Aufbau einer Routing-Tabelle:**

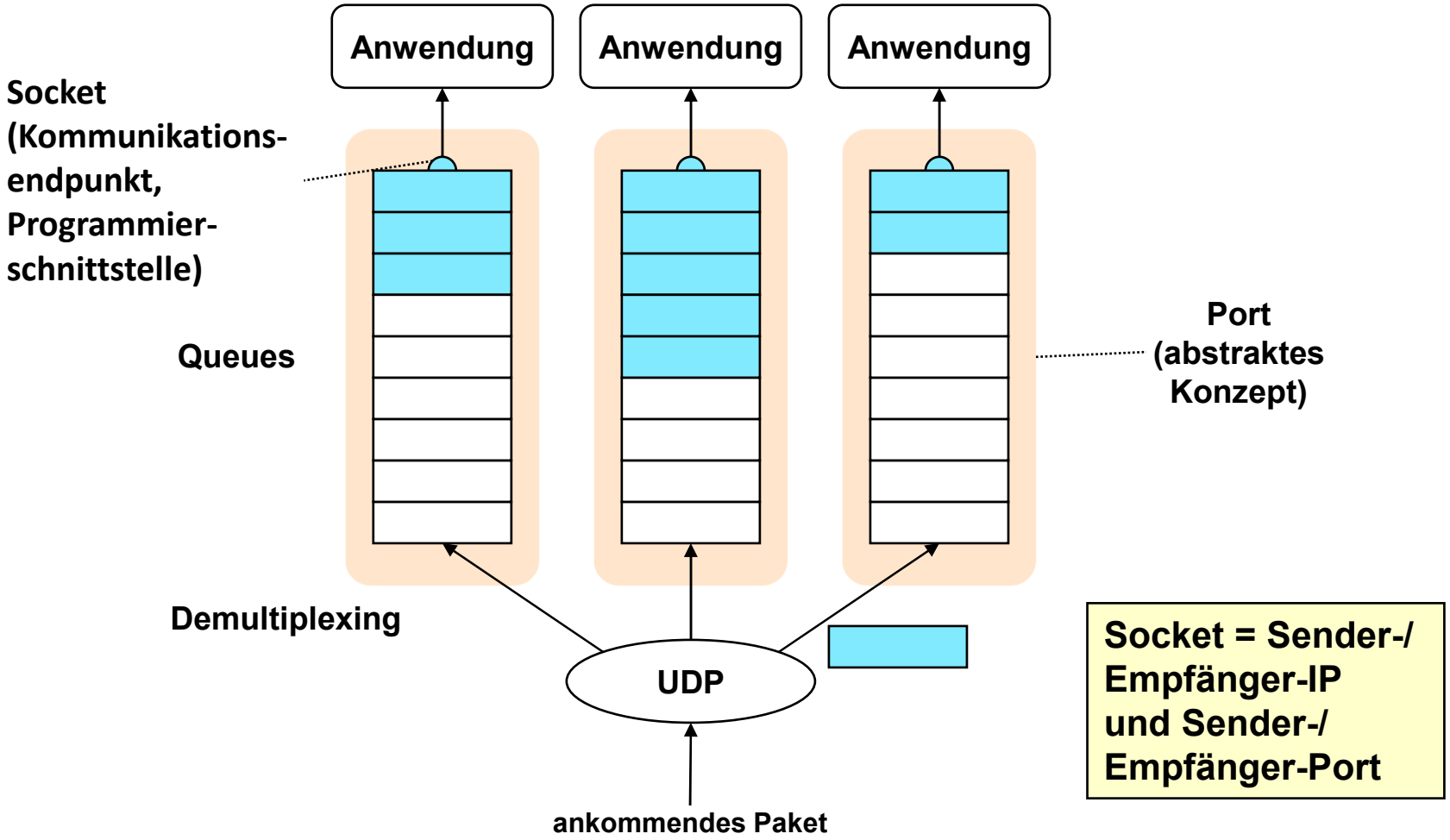
⇒ **dynamisches Routing d.h. Routing auf der Basis von Routing-Protokollen**

- **Einsatzgebiet: in größeren Netzen d.h. in Netzen mit mehreren/ vielen Routern**
- **Topologieänderungen werden automatisch erfasst, aber es gibt eine System-/ Netzwerkbelastung durch die Routing-Protokolle**
- **wichtige Protokolle:** Routing Information Protocol (RIP), Open Shortest Path First (OSPF), Interior Gateway Routing Protocol (IGRP)

***Wann und wo werden die Routing-Protokollinformationen übertragen?***

***Die Routinginformationen werden in regelmäßigen zeitlichen Abständen über die gleichen Interfaces und Netzverbindungen übertragen wie die Nutzdaten, quasi im Hintergrund***

# UDP nur Multiplexing/ Demultiplexing (Ports und Sockets) d.h. Adressierung der Anwendung




Socket = Sender-/ Empfänger-IP und Sender-/ Empfänger-Port

# Überblick TCP

**Grundeigenschaften:**

- **Punkt-zu-Punkt-Verbindung**
- **Streaming-Schnittstelle**
  - Byteorientiert
- **Zuverlässige, verbindungsorientierte Übertragung**
  - **Drei Phasen**: Verbindungsaufbau, Nutzdatenübertragung, Verbindungsabbau
  - Zuverlässiger Verbindungsaufbau, Initialisierung des Empfänger/Sender-Kontexts (→ *Three-Way-Handshake*)
  - Zuverlässige Nutzdatenübertragung (→ *Retransmission*)
  - zuverlässiger Verbindungsabbau
- **Eine Vollduplex-Verbindung**
  - bidirektionaler Datenfluss in derselben Verbindung
  - Optimale maximale Segmentgröße (512-1500 Byte) beachten
- **Flusskontrolle** und **Überlastkontrolle**

# Verfahren zur Autokonfiguration

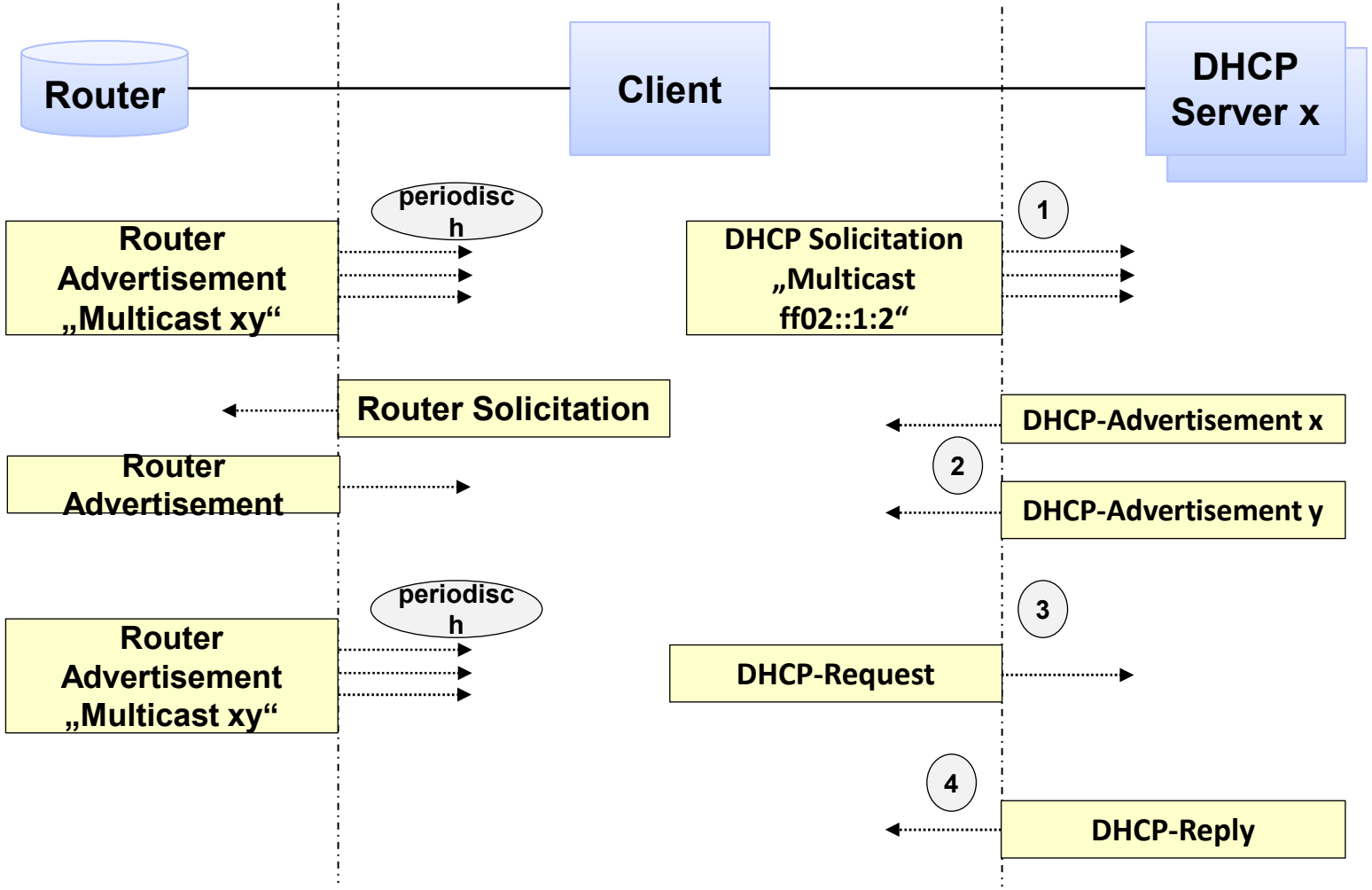
- **Bei IPv4 stehen zwei Verfahren zur Autokonfiguration zur Verfügung:**
  - (Das Bootstrap Protocol (BOOTP) für die Konfiguration von Terminals und festplattenlosen Workstations)
  - **Das Dynamic Host Configuration Protocol (DHCPv4), um Clients eine Netzwerkkonfiguration zuzuweisen**
- **Bei IPv6 unterscheidet man grundsätzlich zwei Methoden zur Autokonfiguration:**
  - **Stateless Address Autoconfiguration (SLAAC)**

    „Stateless“ bedeutet, dass bei der Autokonfiguration kein „Status“ (= IPv6-Adresse) gespeichert wird d.h. ein Client generiert die IPv6-Adresse eigenständig. Für die Konfiguration wird also kein spezieller Server benötigt, was dieses Verfahren in der Administration besonders einfach macht
  - **Stateless/Stateful Address Autoconfiguration (mit DHCPv6)**

    Für dieses Verfahren wird ein DHCPv6-Server benötigt, der die Konfigurations-informationen für die Systeme im Netz speichert und verwaltet. Hierdurch hat der Netzadministrator bessere Möglichkeiten, das Netz zu steuern und zu überwachen (Stateful => IPv6-Adresse wird an Client verteilt)
- **Die beiden Verfahren SLAAC und DHCPv6 können auch kombiniert werden.**

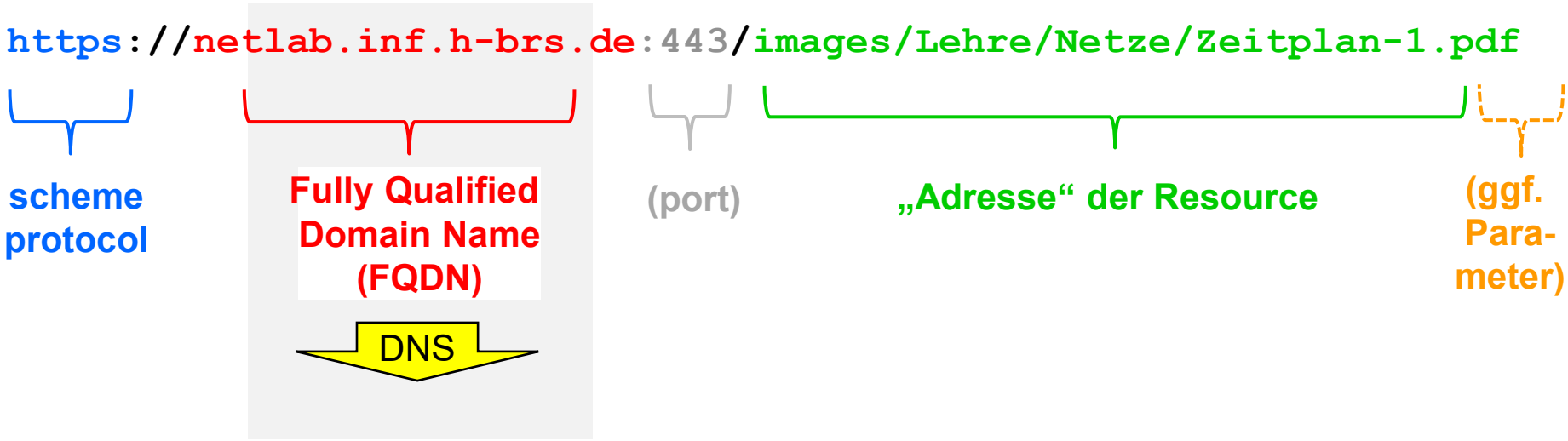
  Zum Beispiel SLAAC für globale IPv6-Adresse und Default Route, die weiteren Parameter über DHCPv6

# Zusammenfassung: SLAAC und DHCPv6 unter Verwendung von Multicast statt Broadcast bei der IPv6-Autokonfiguration



# DNS Namensraum: URI - URL - Domain

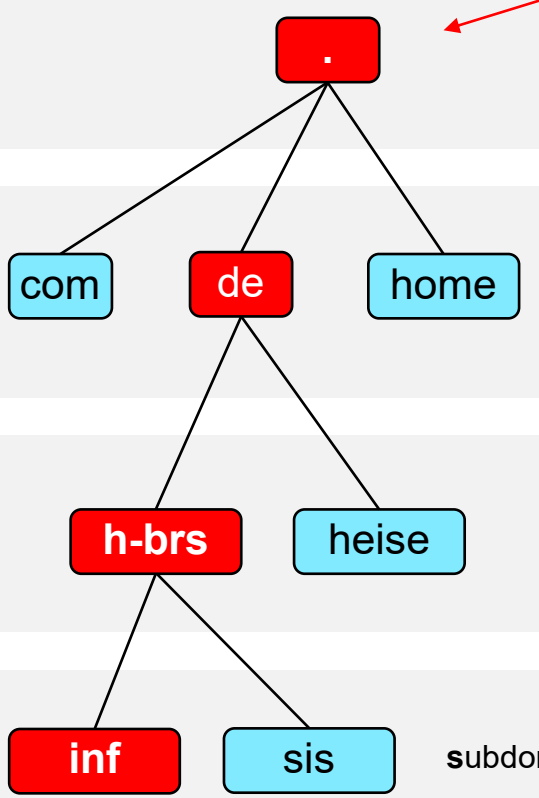
- Jede Ressource im Internet wird kann durch einen vom Menschen lesbaren Identifikator identifiziert werden
- URI = Uniform Resource Identifier (siehe auch RFC 3986)
- URL = Uniform Resource Locator (häufig gleich verwendet, laut RFC 3986 URL = URI in einem Computernetzwerk)
- Grundprinzip, eingeführt 1994 durch Timothy John Berners-Lee (Erfinder des WWW)



# Struktur der Domain Names

Der letzte „Punkt“ wird normalerweise weggelassen

DNS Root




Die Rootdomain ist leer

TLD, top level domain

- Generische Domänennamen (gTLD) z.B. com, net
- CountryCode-Domänennamen (ccTLD)
- Private TLDs, z.B. home/corp/mail (seit 2018)

Internet Corporation for Assigned Names and Numbers (ICANN) koordiniert TLDs und DNS

**S**LD, second level domain

- Werden in der Regel von einem beim zuständigen NIC (denic.de) akkreditierten privatwirtschaftlichen Registrar (Internet Service Provider) verkauft und betreut

**s**ubdomain

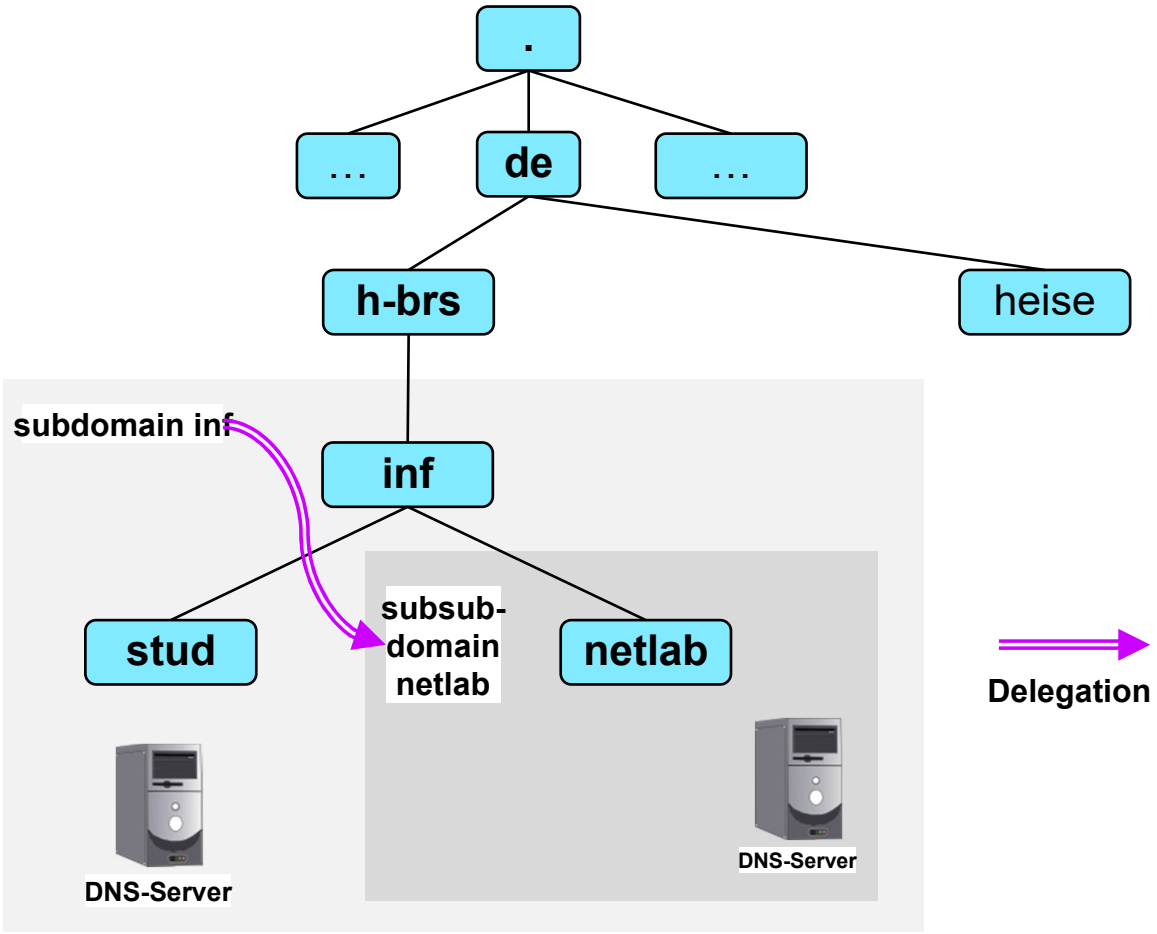
- Werden durch den Kunden verwaltet

**z.B. Fully Qualified Domain Name: netlab.inf.h-brs.de.**



NIC = Network Information Center

# Domäne, Delegation, Zone und Zone-File

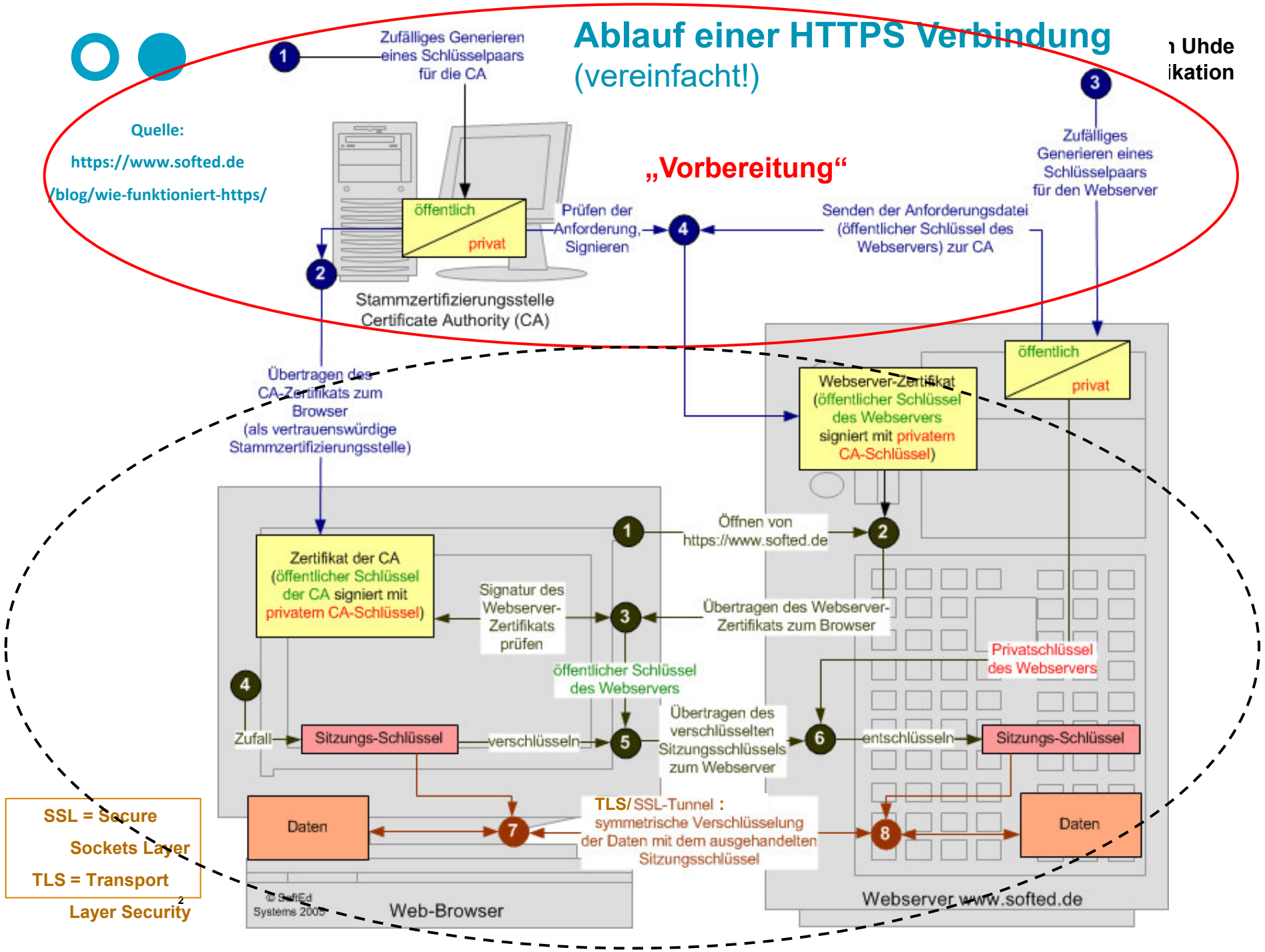



- Die technische und organisatorische Verantwortlichkeit („Authority“) für eine Domäne (Subdomäne, …) kann an einen anderen Administrator übergeben werden („Delegation“).
- Ein Administrator nutzt einen DNS-Server, um den Bereich, für den er verantwortlich ist zu verwalten („Zone“).
- Ein DNS-Server wird durch einen Zone-File konfiguriert, der alle notwendigen Konfigurations-informationen enthält. Findet eine Delegation statt, so muss auch diese im Zone-File beschrieben werden.

Eine Domäne kann in mehrere Zonen aufgeteilt werden. Jede Zone wird durch Name-Server (NS) verwaltet. Das Zone-File enthält „Ressource-Records“ z.B. Namen und die IPv4- bzw. IPv6 Adresse oder einen NS.

# Ablauf einer HTTPS Verbindung
(vereinfacht!)

Quelle:
https://www.softed.de
/blog/wie-funktioniert-https/




SSL = Secure Sockets Layer
TLS = Transport Layer Security

# SSH – Protokollablauf vereinfacht (Client/Server-Szenario)

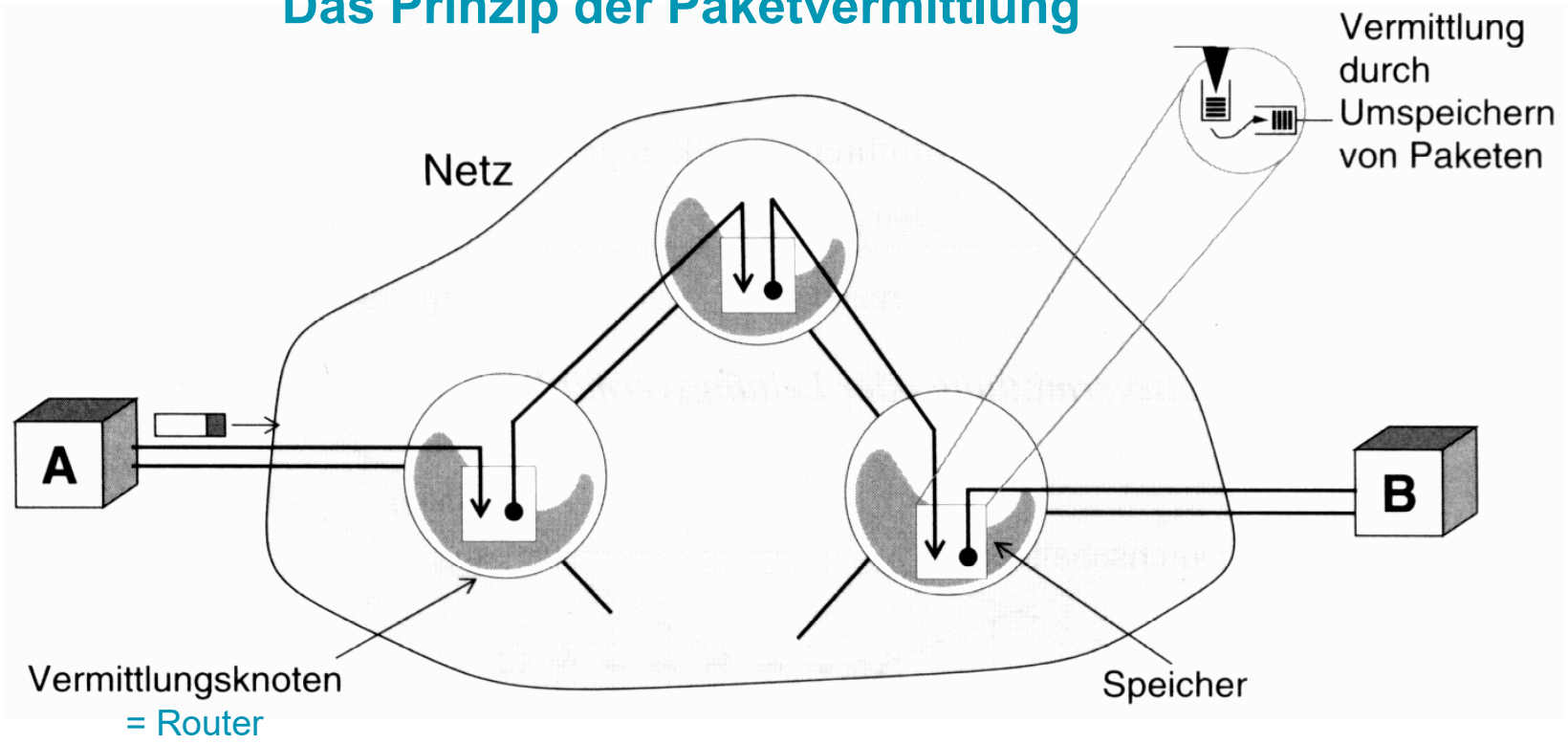
**Voraussetzung: Server besitzt asymmetrisches Schlüsselpaar (Key-$S_{pub}$, Key-$S_{sec}$)**

- **Schritt 1 – Client:**
  - TCP-Verbindung aufbauen
  - Protokollversion austauschen
- **Schritt 2 - Server:**
  - sendet öffentlichen Server-Key Key-$S_{pub}$ und
  - Liste der unterstützte Verschlüsselungsalgorithmen
- **Schritt 3 - Client:**
  - akzeptiert Server-Key (bereits bekannt bzw. Zertifizierungsstelle fragen)
  - wählt Verschlüsselungsalgorithmus
  - generiert symmetrischen Session-Key
  - sendet diesen Session-Key verschlüsselt mit öffentlichem Server-Key
- **Schritt 4 - Server:**
  **entnimmt Session-Key und schaltet auf Verschlüsselung um.**
- **Bei SSH: Schritt 5 - Client:**
  **authentifiziert sich in geeigneter Weise**
  **(das ist der entscheidender Punkt! Es könnte ja jeder kommen.)**

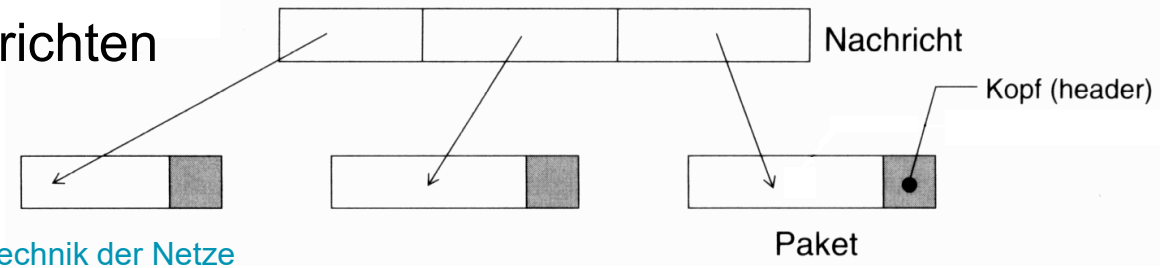
# SSH – Protokollablauf vereinfacht

- **Methoden der Clientauthentifizierung**
  - **User-Passwort: User loggt sich über SSH als user mit Passwort PW-S ein**
  - **Authentifizierung über öffentliche Schlüssel (ähnlich digitale Unterschrift):**
    - **User erzeugt im Vorfeld ein asymmetrisches Schlüsselpaar ($Key\text{-}U_{pub}$, $Key\text{-}U_{sec}$) und hinterlegt im Vorfeld seinen öffentlichen Schlüssel $Key\text{-}U_{pub}$ beim Server.**
    - **Der Client verschlüsselt den öffentlichen Server-Schlüssel $Key\text{-}S_{pub}$ mit dem geheimen Userschlüssel $Key\text{-}U_{sec}$. und sendet das Ergebnis**
      $$ERG = enc_{Key\text{-}Usec}\ (Key\text{-}S_{pub})$$
      **an den Server.**
    - **Der Server besitzt den öffentlichen Schlüssel $Key\text{-}U_{pub}$ des Users. Mit diesem entschlüsselt er das Ergebnis ERG . Kommt als bei der Entschlüsselung $Key\text{-}S_{pub}$ heraus, weiß er, dass der Client den echten geheimen Schlüssel besitzt.**

# Das Prinzip der Paketvermittlung

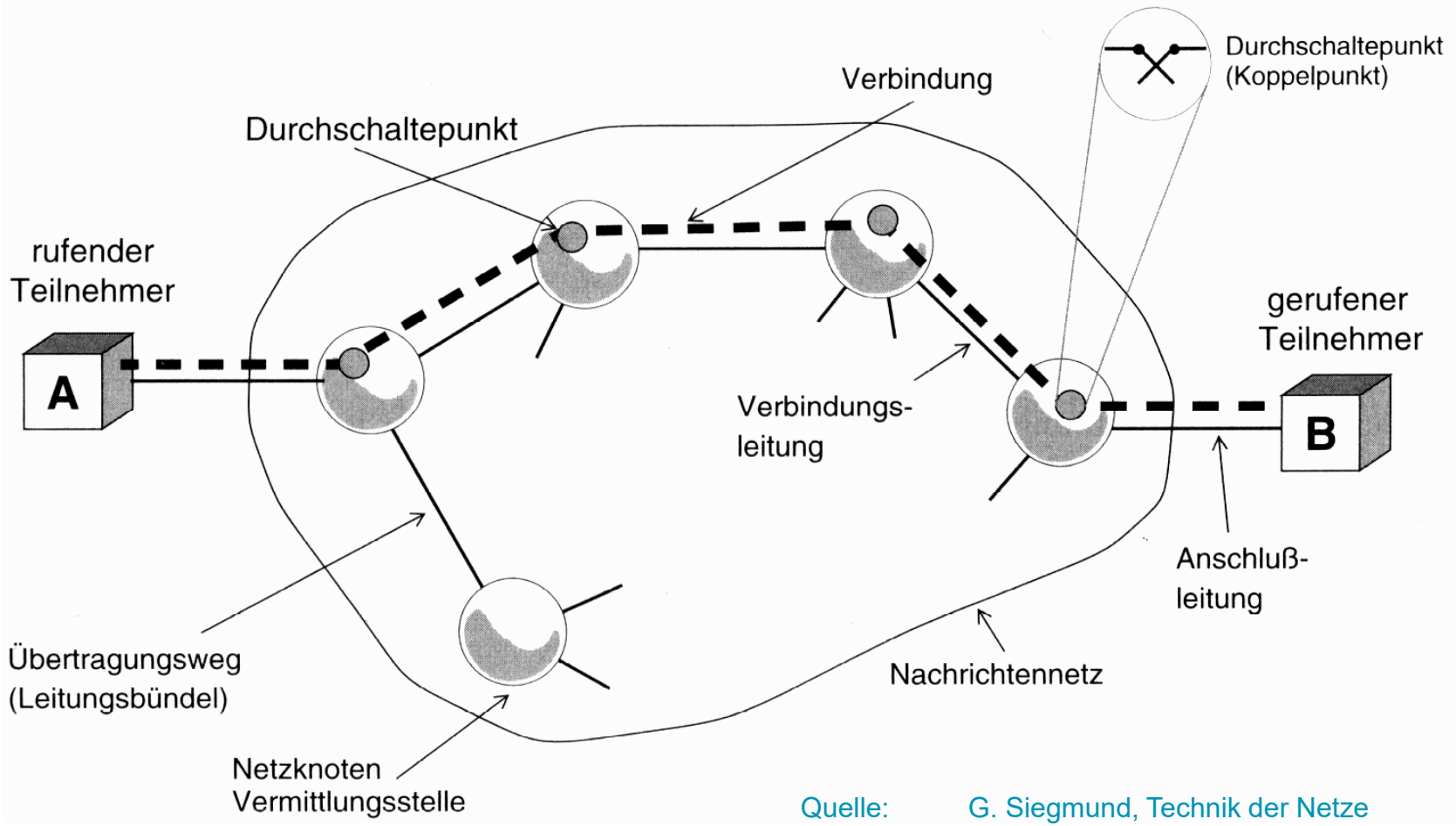



## Aufteilung der Nachrichten in Pakete



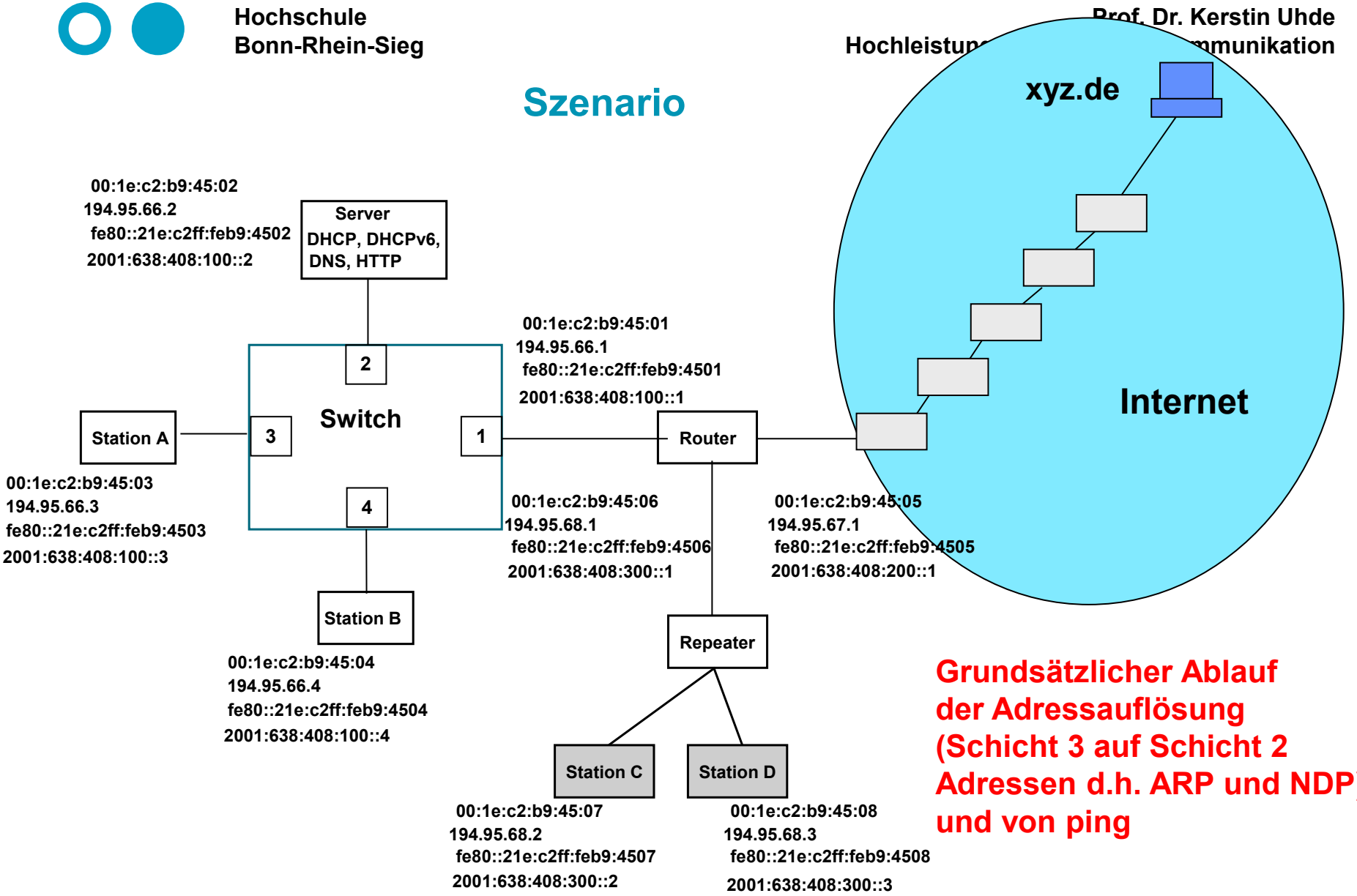

Quelle: G. Siegmund, Technik der Netze

# Das Prinzip der Leitungsvermittlung




Quelle: G. Siegmund, Technik der Netze

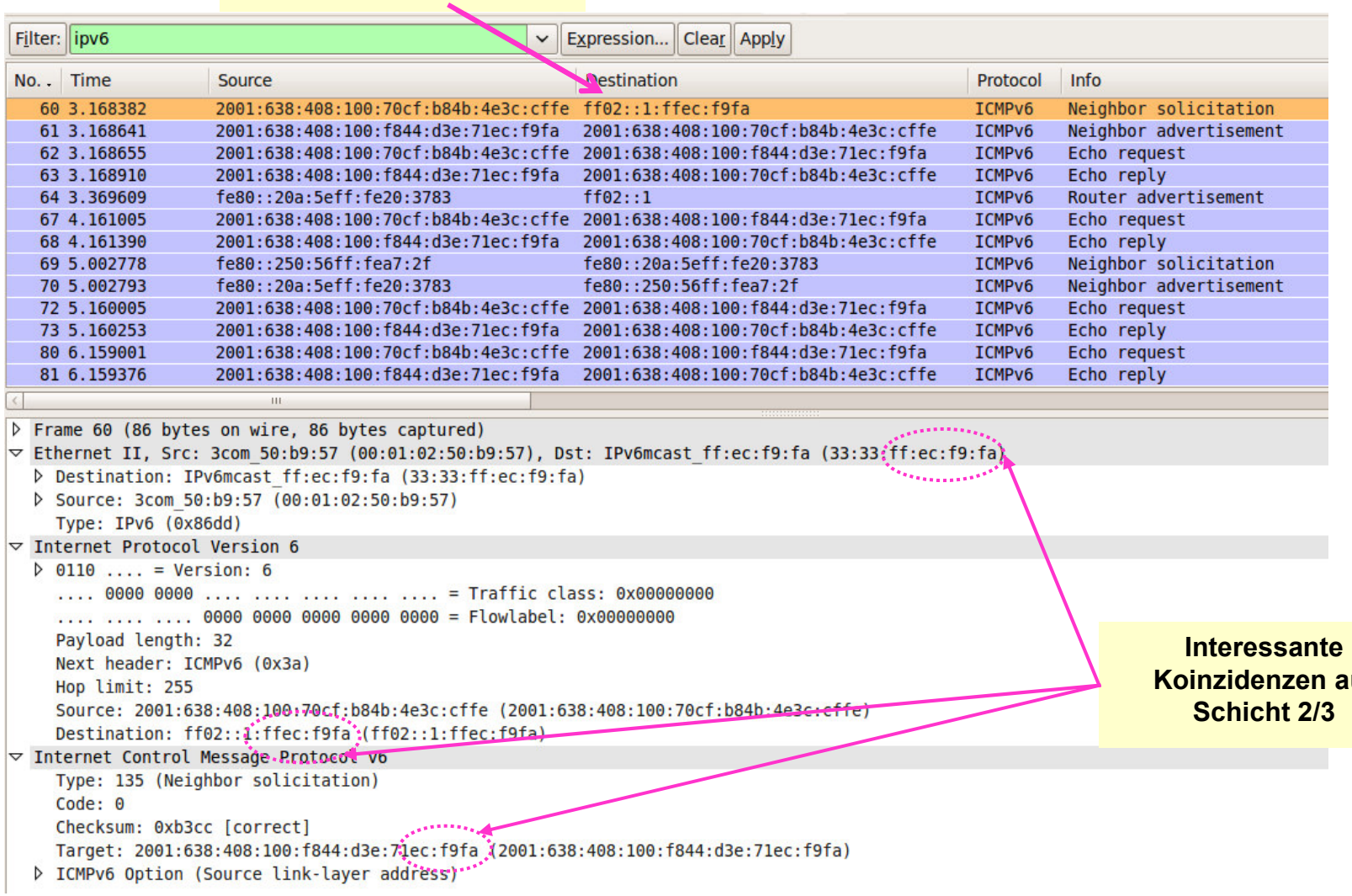
# Szenario




**Grundsätzlicher Ablauf der Adressauflösung (Schicht 3 auf Schicht 2 Adressen d.h. ARP und NDP) und von ping**

# ICMPv6 Neighbor Solicitation + Advertisement

**IPv6-Multicast (ff02:: …**




**Interessante Koinzidenzen auf Schicht 2/3**