DIHSÜD
Digital Innovation Hub

DIGITALISIERUNG FÜR KMU MÖGLICH MACHEN

DER DIGITAL INNOVATION HUB SÜD ALS KOSTENLOSES SERVICE FÜR KMU

Der DIH SÜD
wird unterstützt von:

FFG Forschung wirkt.
Das Land Steiermark
LAND KÄRNTEN

DIHSÜD

FH | JOANNEUM
University of Applied Sciences

Modul
Networking

Networking

Mathias Knoll

Onedrive Share

Basistraining Penetration Testing

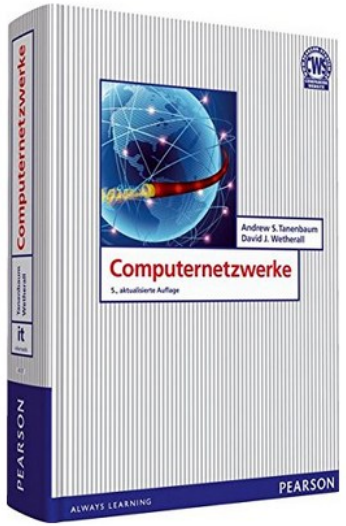
Literatur

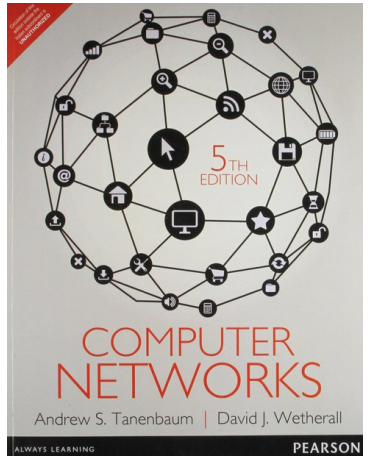
- Computernetzwerke *(Hauptquelle!)*
  Andrew S. Tanenbaum, Pearson Studium
  ISBN 3-8273-7011-6

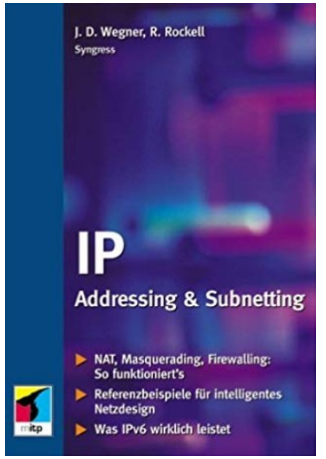
- IP Addressing & Subnetting
  J.D. Wegner, R. Rockell, MITP-Verlag
  ISBN 3-8266-4077-2

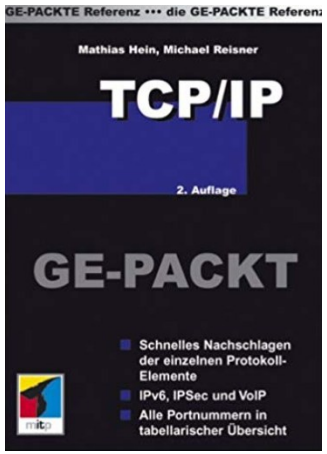
- TCP/IP-Grlg. - Protokolle und Routing
  Gerhard Lienemann, Heise Verlag
  ISBN 3-88229-180-X

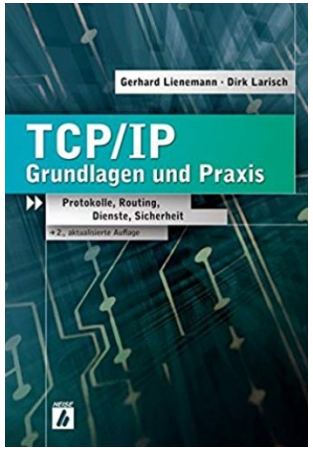
- TCP/IP GE-PACKT *(eher zum Nachschlagen)*
  Mathias Hein, MITP-Verlag
  ISBN-10: 3826613961











Onedrive Share

Internet

- **Institute of Electrical and Electronics Engineers**
  https://www.ieee.org/index.html

- **IEEE 802 LAN/MAN Standards Committee**
  http://www.ieee802.org/
  http://www.ieee802.org/3

- **Request for Comments (RfC) der Internet Engineering Task Force (IETF)**
  https://www.ietf.org/rfc.html

- **FTP Server mit Kursunterlagen** (und mehr)
  ftp://mesen.fh-joanneum.at/common/knoll/network_technologies/basisausbildung_pen_testing/

Onedrive Share

DIHSÜD

FH JOANNEUM
University of Applied Sciences

Geschichte

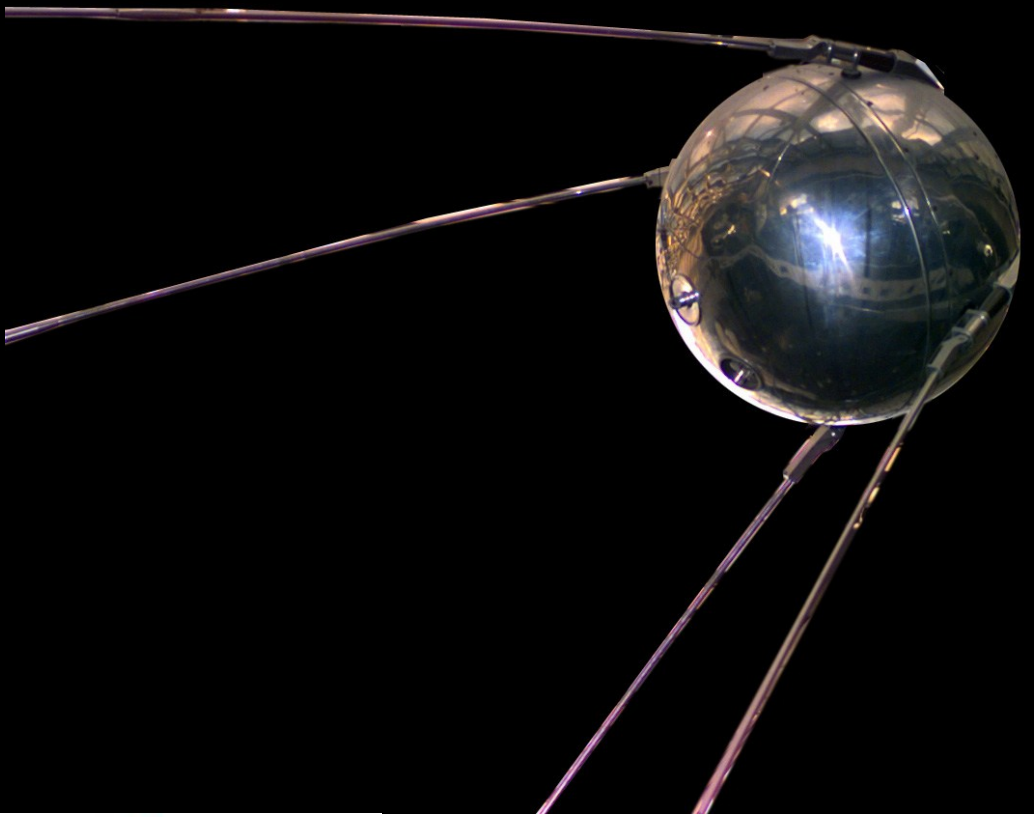
Onedrive Share

Sputnikkrise

ARPA

ARPAnet

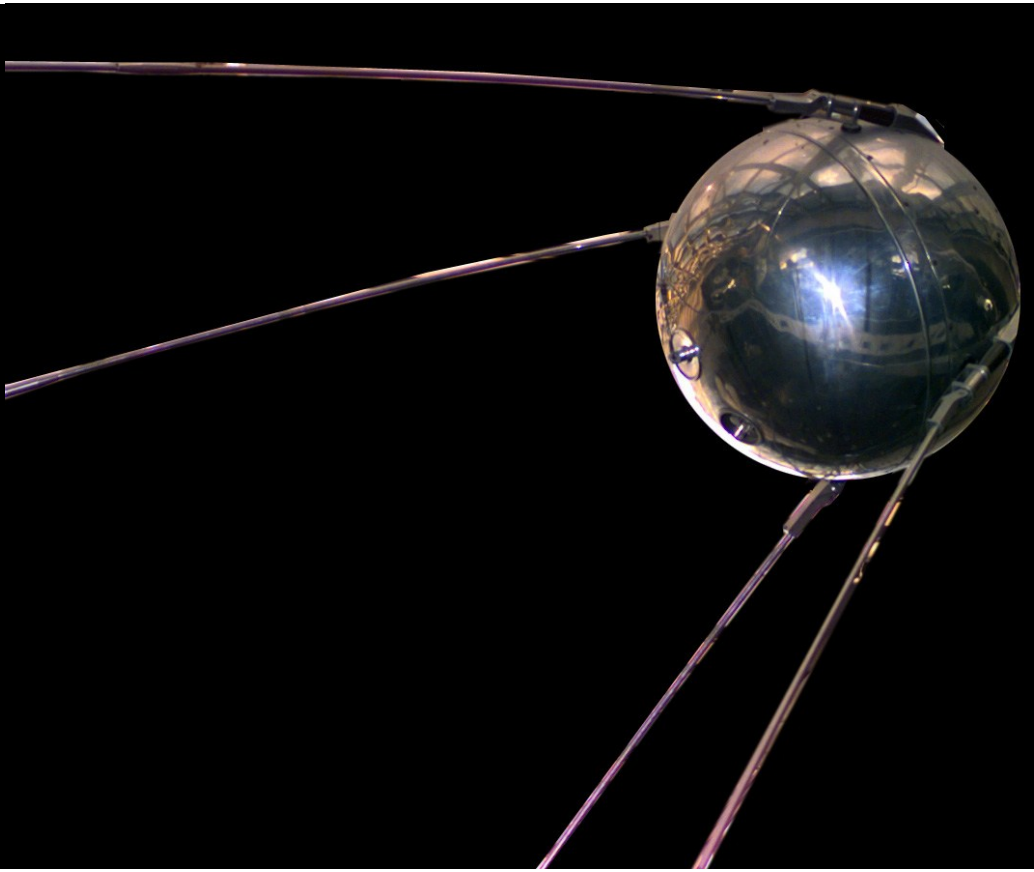
FH | JOANNEUM
University of Applied Sciences



Was **ist** das?
Was hat es mit dem **Internet** zu tun?

DIHSÜD

Sputnik | 1957



turday Evening, October 5, 1957 (UP)—Means United Press Price: Five

Russians Win Race To Launch Earth Satellite

Man On Threshold Of Space Travel

By DANIEL F. GILMORE
United Press Staff Correspondent

LONDON (UP)—The pulsating radio "beep" of the first manmade earth satellite signalled today to the world that man had crossed the threshold into the age of travel through space.

The Soviet Union announced it had won the race into space by launching an earth satellite Friday, a 184-pound, 22-inch globe now orbiting the earth at 18,000 miles an hour, 560 miles up.

Millions of persons throughout the world heard the "beep...beep... beep..." rebroadcast today by local stations and realized that man had taken his first faltering steps into the new era.

Launching of the satellite was a tremendous victory for science. It was a more tremendous victory for Soviet propaganda to be able to trumpet to the world the Russians were the first to break through the frontiers of space.

Bolsters ICBM Claims

— WEATHER —

WEST VIRGINIA—Partly cloudy with highest in the 60s today and Sunday. Lowest tonight 50 west and 40 east portions.

VIRGINIA—Fair with lowest 45 to 50 west and north and 50 to 55 southeast portions tonight, Sunday mostly sunny and a little warmer. Tides on the coast and lower bay will run a foot or two above normal.

How To Spot Satellite

By UNITED PRESS

Here's how to look for the Russian earth satellite which will be whizzing through the sky at 18,000 miles an hour.

The best time to spot it is at dawn or dusk when the sky is semi-dark. There is a chance that it could be seen if it travels across the face of the moon at night.

The best instruments to use are ordinary binoculars or telescopes. Powerful telescopes won't pick it up because of their narrow fields.

Through optical instruments, the satellite will look like the faintest star which can be seen with the naked eye.

Keep a sharp eye out. The satellite travels so fast it may appear on the horizon for only seconds and chances of spotting it have been estimated at one in a hundred.

U. S. May Spe Up Satellite Program

By JOSEPH L. MYL
United Press Staff Corre

WASHINGTON (UP)—A scientists caught fastboo Russia's epic launching of man-made moon, indicate the United States may s its own earth satellite prog

Leaders of the U.S. satel gram also said that it Russia rocketed its hea pound satellite into a gl diing orbit with a rocke to an intercontinental missile.

That could mean Rus only has beaten this count frontiers of space, but also has been called the weapon" for modern day ICBM. This country has tested a successful ICBM

American diplomats Russia had scored a note

Die ARPA Direktive | 1958

February 7, 1958
NUMBER 5105.15

Department of Defense Directive

SUBJECT Department of Defense Advanced Research Projects Agency

I. PURPOSE

The purpose of this directive is to provide within the Department of Defense an agency for the direction and performance of certain advanced research and development projects.

II. RESPONSIBILITY AND AUTHORITY

A. Establishment

In accordance with the provisions of the National Security Act of 1947, as amended, and Reorganization Plan No. 6 of 1953, there is established in the Office of the Secretary of Defense the Department of Defense Advanced Research Projects Agency. The Agency will be under the direction of the Director of Advanced Research Projects.

B. Responsibility

The Agency shall be responsible for the direction or performance of such advanced projects in the field of research and development as the Secretary of Defense shall, from time to time, designate by individual project or by category.

C. Authority

Subject to the direction and control of the Director:

1. The Agency is authorized to direct such research and development projects being performed within the Department of Defense as the Secretary of Defense may designate.

2. The Agency is authorized to arrange for the performance of research and development work by other agencies of Government, including the military departments, as may be necessary to accomplish its mission in relation to projects assigned.

3. The Agency is authorized to enter into contracts and agreements with individuals, private business entities,

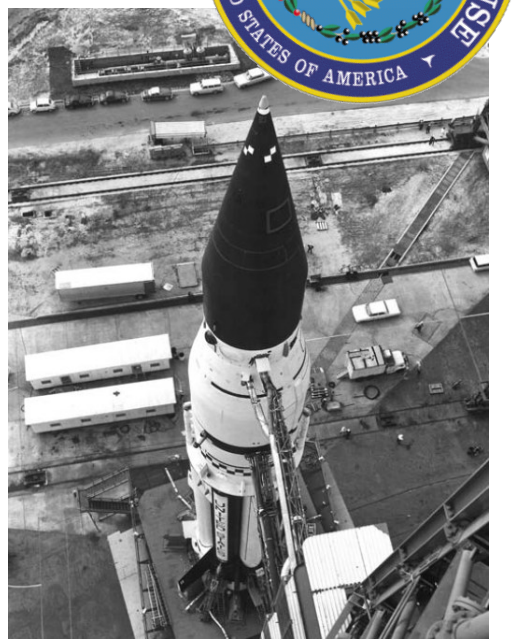
The Advanced Research Projects Agency

Initialer Fokus:

- Raumfahrttechnologie
- Ballistische Raketenverteidigung
- Erkennung von Nuklearwaffentests

The Agency is authorized to enter into contracts and agreements with individuals, private business entities, educational, research or scientific institutions including federal or state institutions.

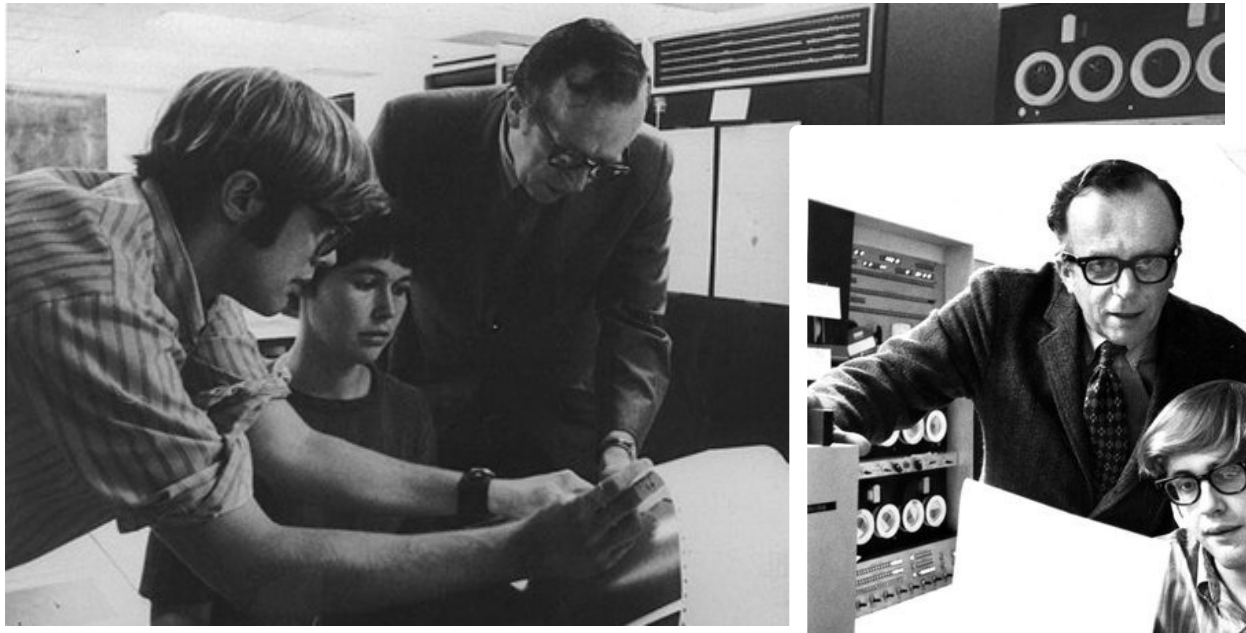
Die Agentur ist **autorisiert**, mit **Individuen, privaten Geschäftseinrichtungen, Institutionen für Lehre, Forschung oder Wissenschaft** aber auch stattlichen Einrichtungen **Verträge abzuschließen.**



Saturn Raketen
mit DARPA Technologie

Information Processing Techniques Office (IPTO) | 1961

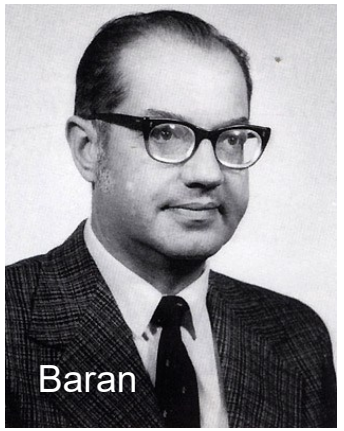
- **Abteilung der DARPA**
- Erster Direktor:
  Joseph Carl Robnett Licklider
  - Visionär
  - Konzept
    "Intergalactic Computer Network"
- **Seine Programme:**
  - **Computer Science Departments in Universitäten,**
  - **Time-Sharing Systeme** und
  - **Netzwerke**

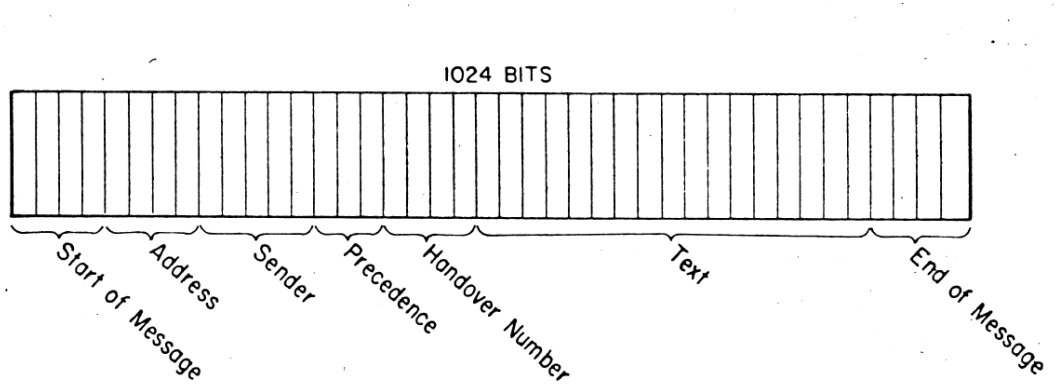


Netzwerke | 1962/3 | "Packet Switching" (Paketvermittlung)

statt Leitungsvermittlung

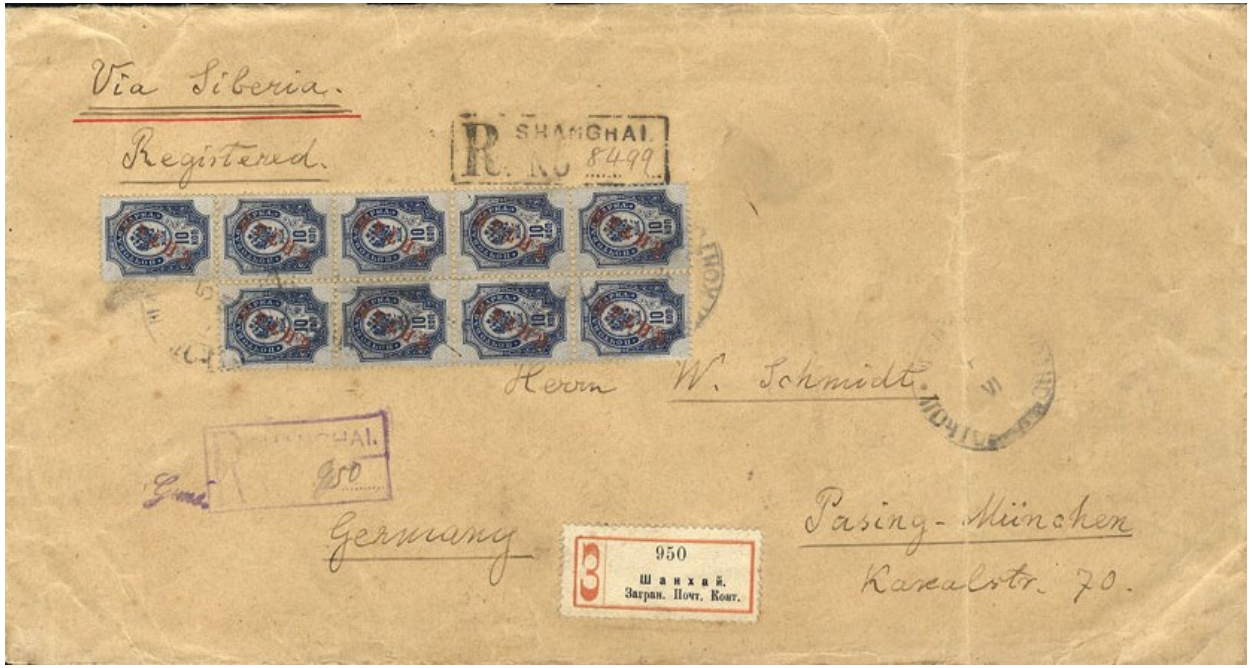
- **Paul Baran** (Rand Corp) "On Distributed Communication Networks" (1962-)
- **Donald Davies** (National Physical Laboratory (United Kingdom) **entwickelte den Terminus "Packet"** (1963)
- Aus welchen Teilen besteht ein Paket:
  - **Quelladresse**,
  - **Zieladresse**,
  - **Länge des Datenteils**,
  - **Nummer**
  - **Art des Pakets** und
  - der **Payload** (Nutzlast) – die **Daten** (oder Text)





Paul Baran "On Distrtibuted Communication Networks" (1962)

Netzwerke | Packet Switching - Analogie: Briefe



Netzwerktopologie | 1967

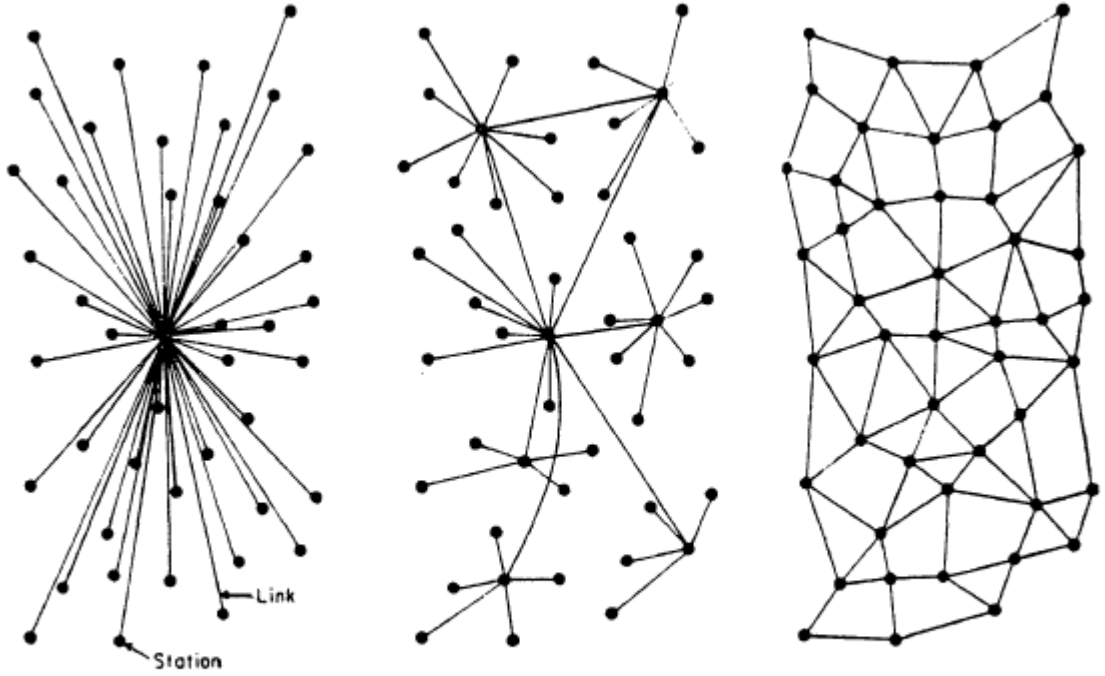
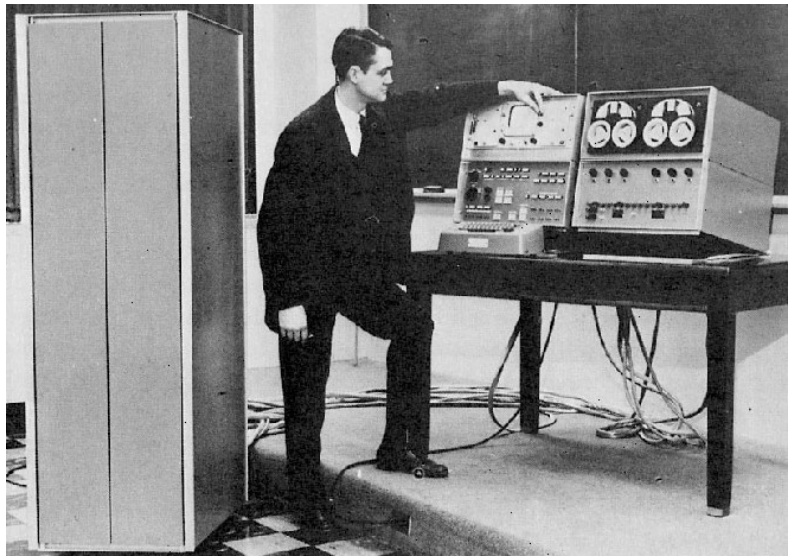


Fig. 1—(a) Centralized. (b) Decentralized. (c) Distributed networks.

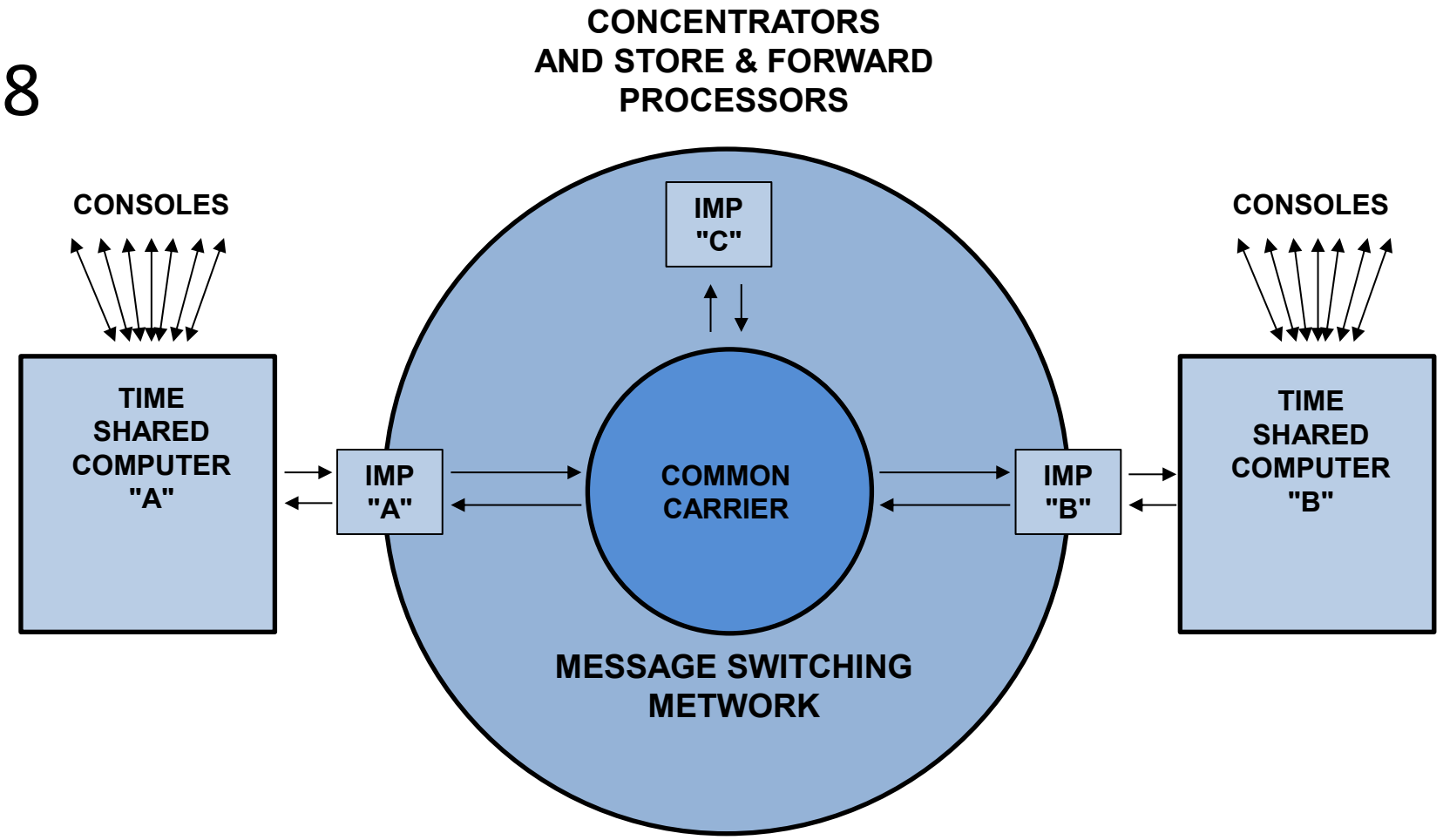
Paul Baran, 1962



Wes Clark, Erfinder des ersten Minicomputers **LINC** (**Laboratory INstrument Computer**) kam **1967** mit der Idee, Minicomputer als Nachrichtenüberbringer einzusetzen- also als **Vermittler zwischen Host und Netzwerk**

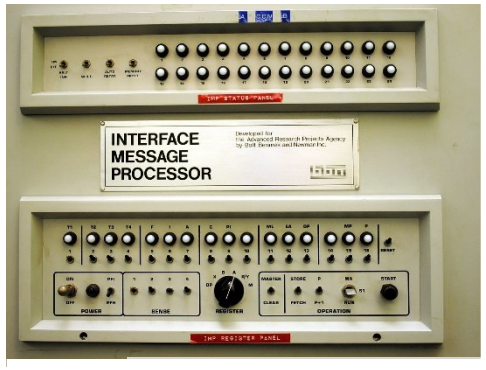
Entwurf | 1968

Larry Roberts (2016)
https://www.youtube.com/watch?v=qkD4HVRnGJE



USE OF INTERFACE MESSAGE PROCESSORS (IMP)
FOR ROUTINNG AND CHECKING INTER COMPUTER COMMUNICATION

Der erste Netzwerkknoten | 1969



Interface Message Prozessor
"Gateway/Router"
Protokoll 1822 *(RFC802)*

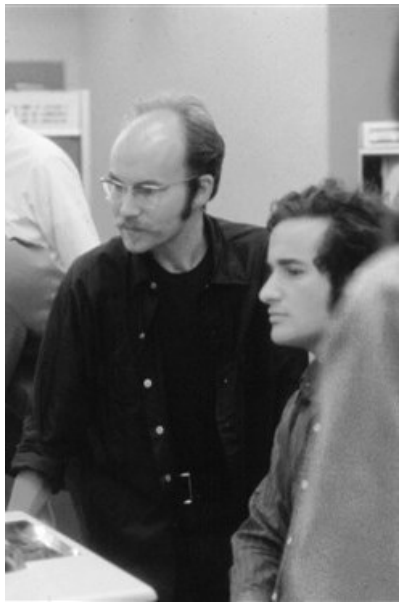
SDS Sigma 7
"32Bit Host"

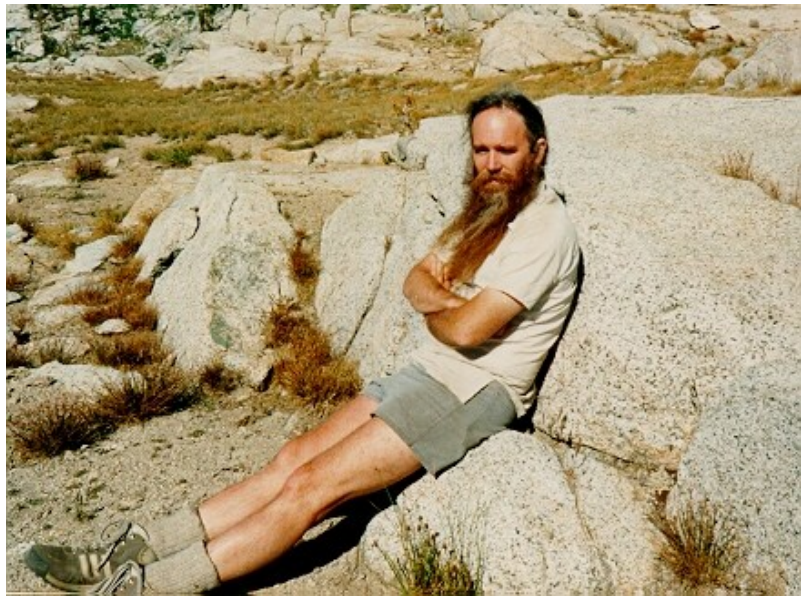
```
   +-+--------------+--------+------+---------------------------+
   |*|   Op Code    |   R    |  X   |    Reference address      |
   +-+--------------+--------+------+---------------------------+
bit 0 1            7 8      1 1    1 1                         3
                            1 2    4 5                         1

Bit   0     indicates indirect address.
Bits  1-7   contain the operation code (opcode)
Bits  8-11  encode a register operand (0:15)
Bits 12-14  encode an index register (1:7). 0 indicates no indexing.
Bits 16-31  encode the address of a memory word.
```

https://en.wikipedia.org/wiki/SDS_Sigma_series

RFCs (Request for Comments) | 1969

"The early R.F.C.'s ranged **from grand visions to mundane details**, although the latter quickly became the most common. Less important than the content of those first documents was that they were available free of charge and anyone could write one. Instead of authority-based decision-making, we relied on a process we called "**rough consensus and running code**." **Everyone was welcome to propose ideas**, and **if enough people liked it and used it, the design became a standard**."





Stephen Crocker, Creator of RFC1, 04/2009
https://www.nytimes.com/2009/04/07/opinion/07crocker.html

RFCs (Request for Comments) | Überblick

- **RFCs? (WTF? Request for Comments!)**

  "We reject kings, presidents and voting. We believe in rough consensus and running code."

  -Dave Clark, MIT / IAB

  - Keine zentrale Organisation für technische Spezifikationen
  - Die Evolution der Internet Spezifikationen ist ein offener, "unkontrollierter" Prozess
  - Jeder "Netizen" kann dran teilhaben, wenn wer eine Anregung hat, reicht er ein RFC ein
  - Basis: RFCs – Request for Comments (http://www.rfc-editor.org/)

- **Über den RFC Editor (und andere Werkezuge) wird der Entwurf kommentiert, eine eindeutige Nummer referenziert darauf**

- **Einmal akzeptiert kann ein RFC nicht mehr geändert, allerdings fortgeführt und ergänzt werden**

- **Einträge:**
  - An Ethernet Address Resolution Protocol (ARP) (RFC826)
  - A Standard for the Transmission of IP Datagrams on Avian Carriers (RFC1149)
  - The Infinite Monkey Protocol Suite (IMPS) (RFC 2795)
  - Electricity over IP (RFC 3251)
  - Management of IP Numbers by Peg-DHCP (RFC2322)
  - usw.

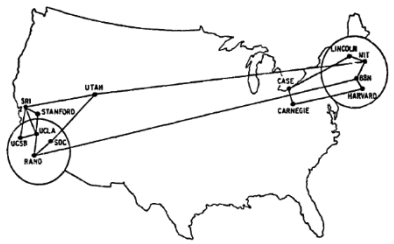
RFC Editor

https://www.rfc-editor.org
https://www.ietf.org/standards/rfcs/

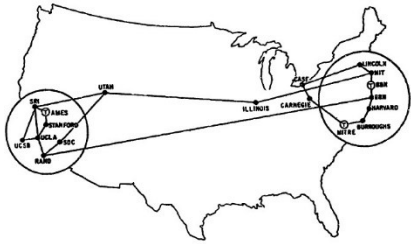
Skalierung und Entwicklungen

ARPAnet
Internet
Protokolle

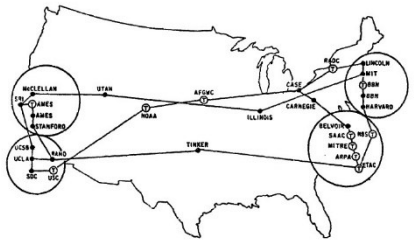
ARPAnet | 1970-1979 | Es skaliert …



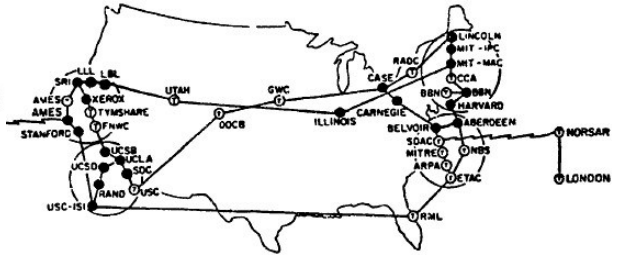
1970



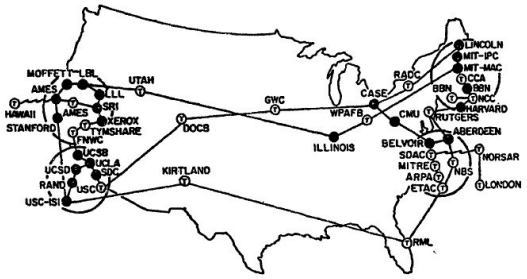
1971



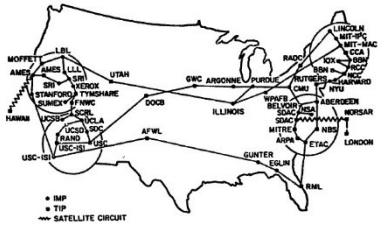
1972



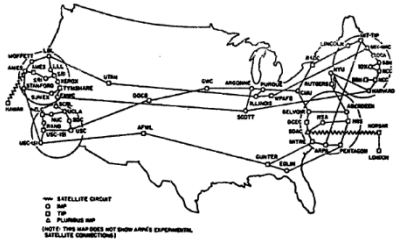
1973



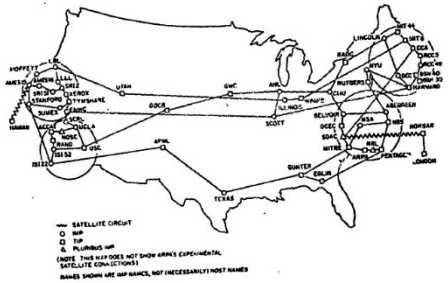
1974



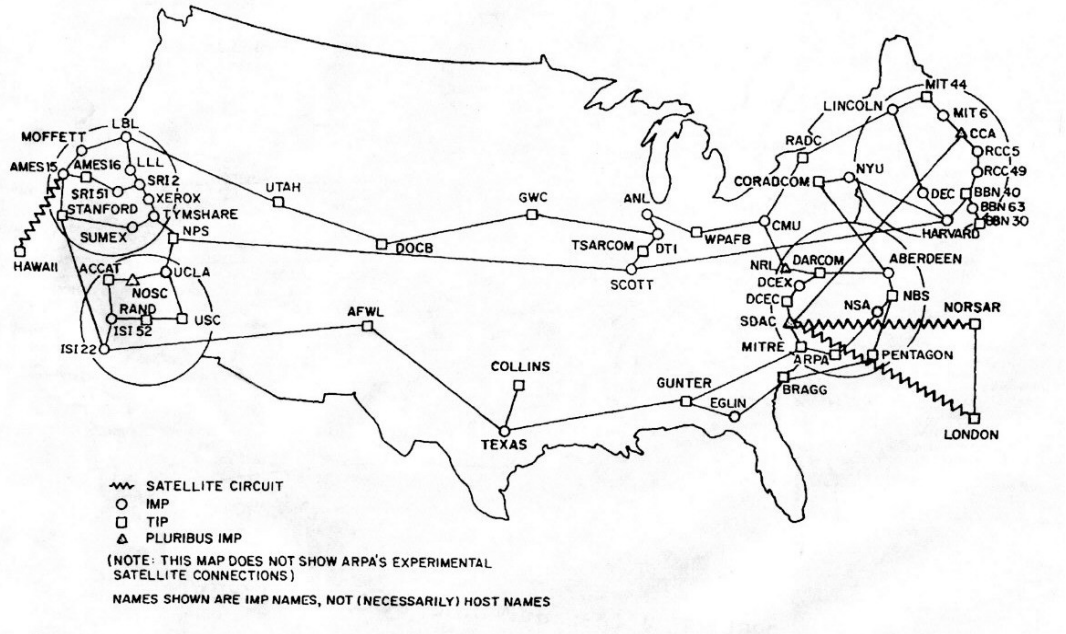
1975



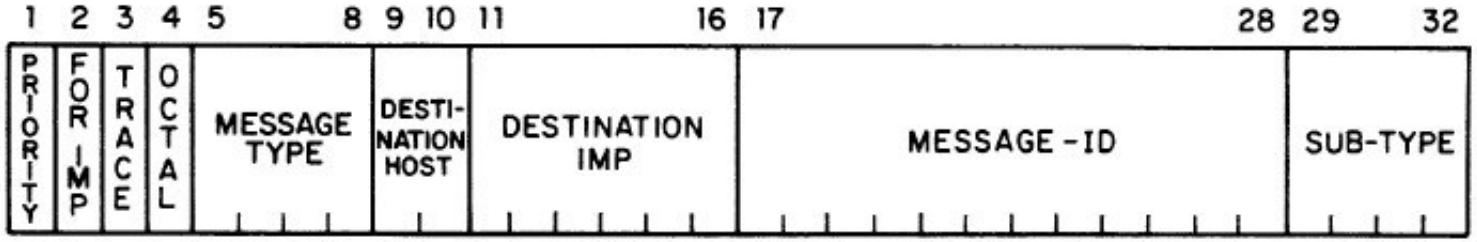
1976



1977

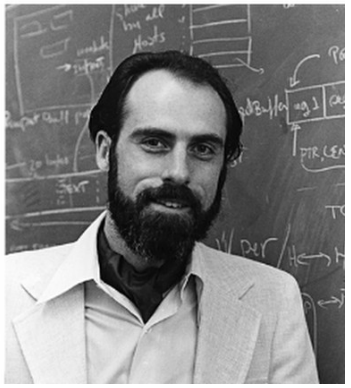


Protokolle | So sprechen Computer übers Netzwerk



Host zu IMP (Teil des **Network Communication Protocols NCP**)

Bob Kahn



Vint Cerf

Bob Kahn und Vint Cerf arbeiten an einem neuen Protokoll, dass die Kommunikation erheblich verbessern soll …

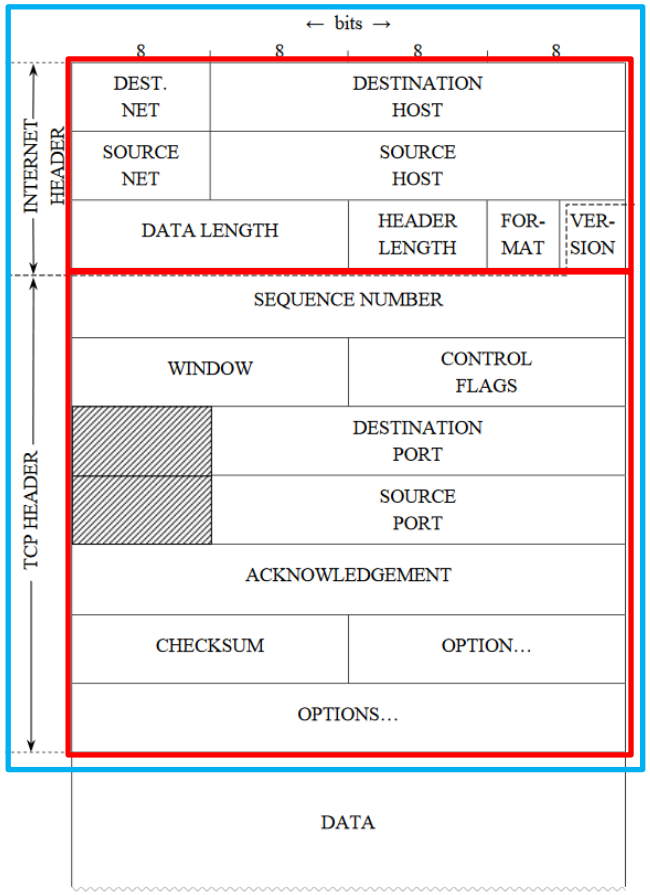
TCP/IP

beide gründen später die

Internet Society

TCP/IP

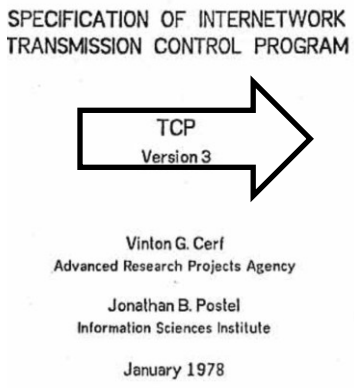
Transmission Control Protocol / Internet Protocol | 1974-1983

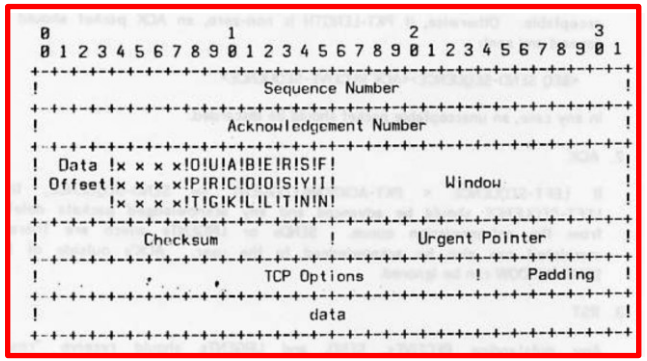


TCPv2 (1974)

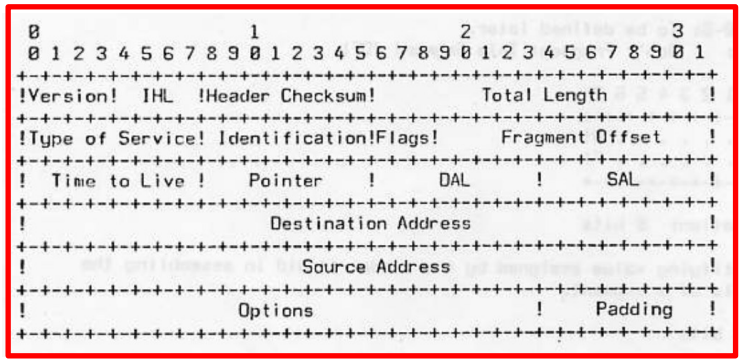
"**We are screwing up** in our design of internet protocols by **violating the principle of layering**. Specifically we are trying to use TCP to do two things: serve as a host level end to end protocol, and to serve as an internet packaging and routing protocol. These two things should be provided in a layered and modular way. I suggest that **a new distinct internetwork protocol** is needed, and that **TCP be used strictly as a host level end to end protocol**."

["Comments on Internet Protocol and TCP", JonPostel,1977 https://www.rfc-editor.org/ien/ien2.txt]





TCPv4 (Juni 1978)



IPv4 (Juni 1978)

1983 - UMSTIEG

Netzwerkmodelle

4 Schichten Netzwerkmodel (TCP/IP)

- Seit 1973
- Implementierung vor Spezifikation
- Aus ARPAnet Protokollen entstanden

7 Schichten Netzwerkmodel (OSI model)

- Seit 1983
- Spezifikation vor Implementierung - Referenzmodel
- Standardisierung durch die Open System Interconnection
- Wird als heute als Beschreibung der Funktionalitäten verwendet
- Protokolle wie ARP (RFC 826, 1982) lassen sich nicht eindeutig einordnen

Wer regiert/regelt das Internet? | Heute

Grundsätzlich: Niemand – Verschiedene Institutionen bieten Richtlinien und Verfahrensanweisungen. Die meisten davon sind (meist U.S.) NPOs - Non-Profit Organisationen.

- **ICANN (Internet Corporation for Assigned Names and Numbers):**
  - Domains und DNS Routing (.com, .at, .net, …)
  - Dienste und „Ports“

ICANN
https://www.icann.org/

- **IANA (Internet Assigned Numbers Authority)**
  - Untersteht ICANN in Koop. mit IETF
  - IP Adressverteilung

iana
http://www.iana.org/

- **IETF (Internet Engineering Task Force):**
  - Technische Spezifikationen → IAB (Internet Architecture Board) / RFC Editor
  - Loose Organisation mit vielen Arbeitsgruppen für viele Bereiche

IETF
https://www.ietf.org/

- **IEEE (Institute of Electrical and Electronics Engineers)**
  - Größter technische Berufsverband der Welt

IEEE
https://www.ieee.org/

DIHSÜD

FH | JOANNEUM
University of Applied Sciences

Geräte & Funktion

Netzwerkkarten
Hubs & Switches
Router

Netzwerkgeräte

Modem

Einwahl ins Netzwerk

des Internet Service Providers

Kabel

Kupfer, Lichtwellenleiter, etc.

NIC (Network Interface Card)

die Netzwerkkarte

Repeater

Verstärkt/verbessert das elektrische Signal damit kann man eine Leitung verlängern

Hubs

Auch Multiport-Repeater genannt, kopieren/verstärken/verbessern sie das elektrische Signal/ auf mehrere Ausgänge.
Wie ein Stomverteiler.

Switches

leiten zielgerichtet Datenpakete an Netzwerkgeräte weiter

Router

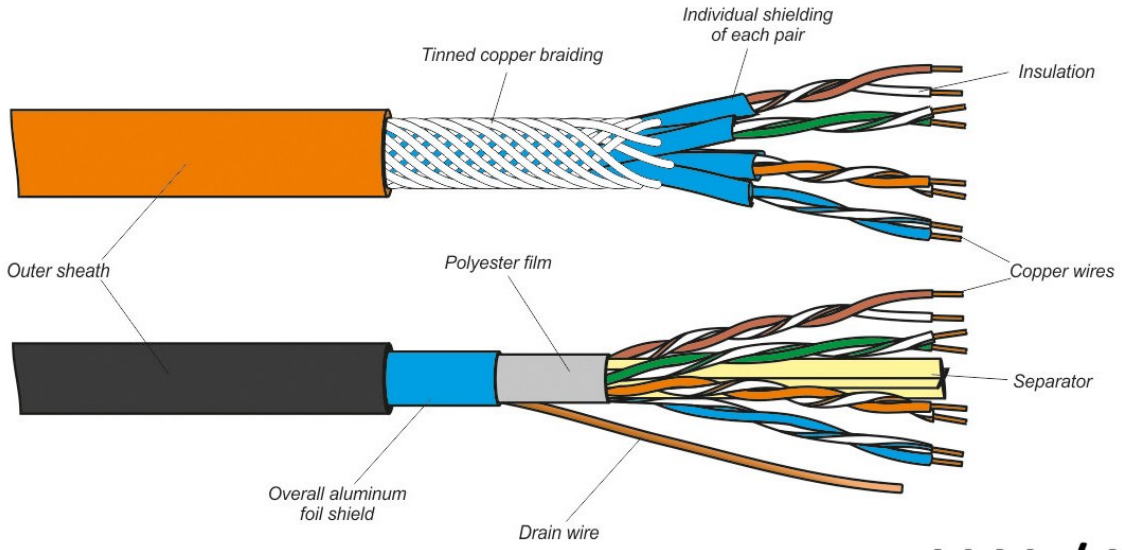
Entscheidet, in welches Netzwerk ein Datenpaket geleitet werden soll

mindestens

Netzwerkkomponenten haben einen "Port"

… die Buchse für den Netzwerkstecker

RJ45



http://www.nikomax-global.com/pubs/articles/46/

X X / X X X

Cable shielding:
U – unshielded
F – foil
S – braid screen
SF – braid screen + foil

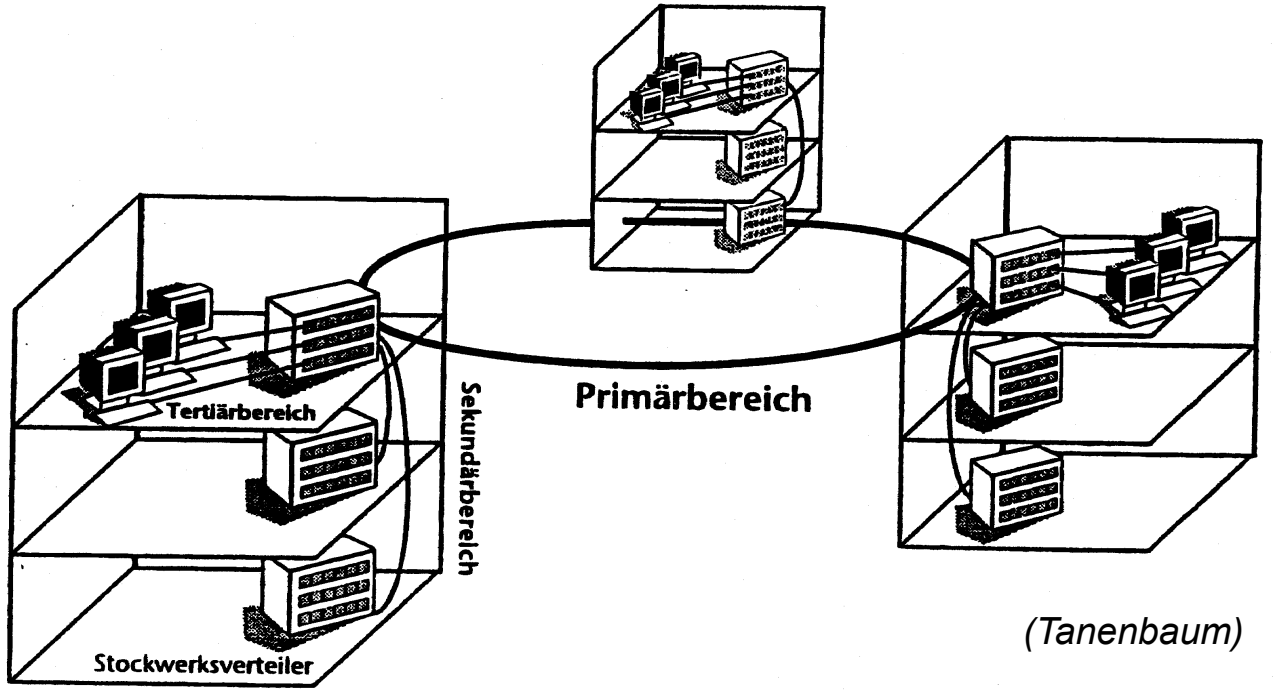
Shielding of each pair:
U – unshielded
F – foil

TP - Twisted Pair

„RJ“ für **Registered Jack** (‚genormte Buchse‘)

Strukturierte Verkabelung

- **Primärverkabelung**
  - Zwischen Gebäuden
  - Vorwiegend Glasfaser
- **Sekundärverkabelung**
  - Etagen von Gebäuden
  - Switches als "Konzentratoren*"
  - Kupfer (TP**-Kabel) oder Glasfaser
- **Tertiärverkabelung**
  - Switches als "Konzentratoren*"
  - Kupferkabel (TP-Kabel)

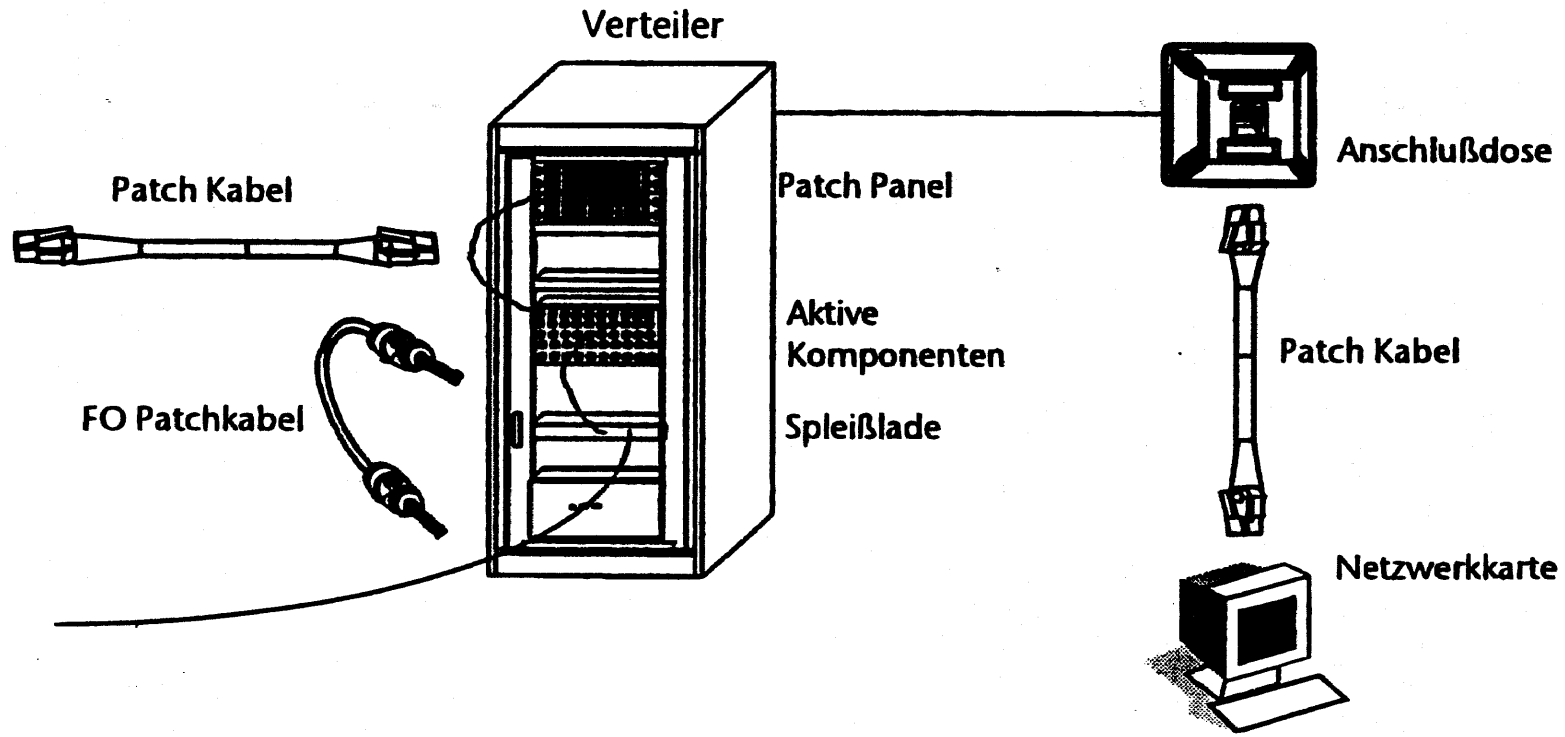


(Tanenbaum)

* Ein Konzentrator ist ein Gerät, das mehrere Leitungen einer bestimmten Datenrate
auf eine Leitung mit größerem Durchsatz zusammenfasst und umgekehrt.

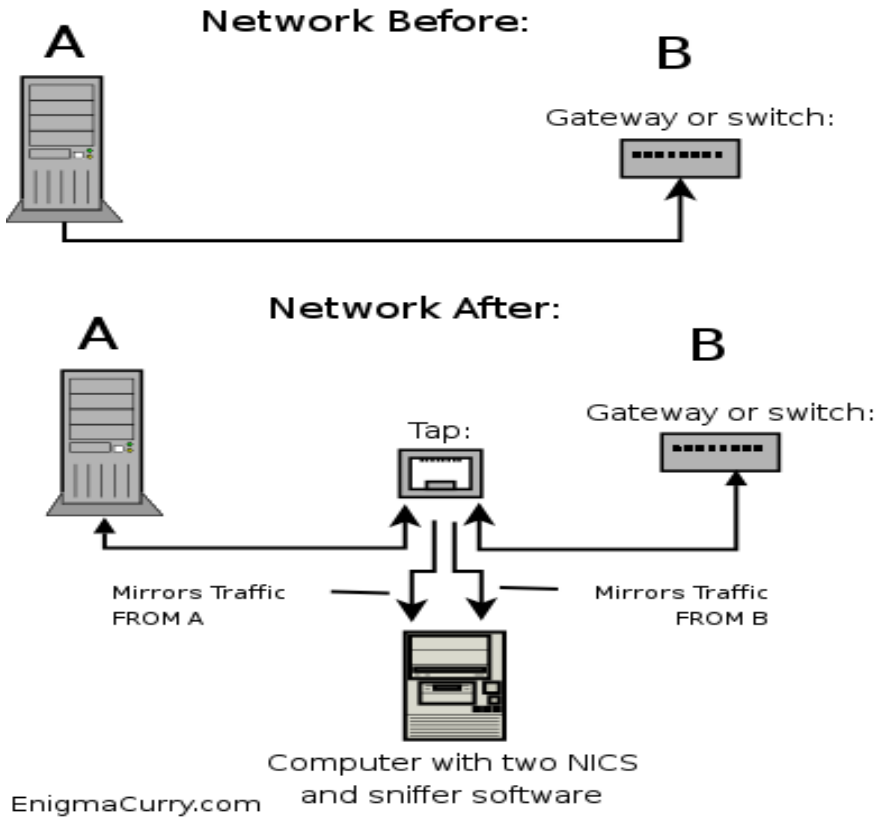
**Twisted-Pair Kabel – d.h. verdrillte Adernpaare im Kabel: Vorteil: Weniger Störungen durch Spannung – Robuster gegen physische Einwirkung- Ziehen,
Drehen, usw.

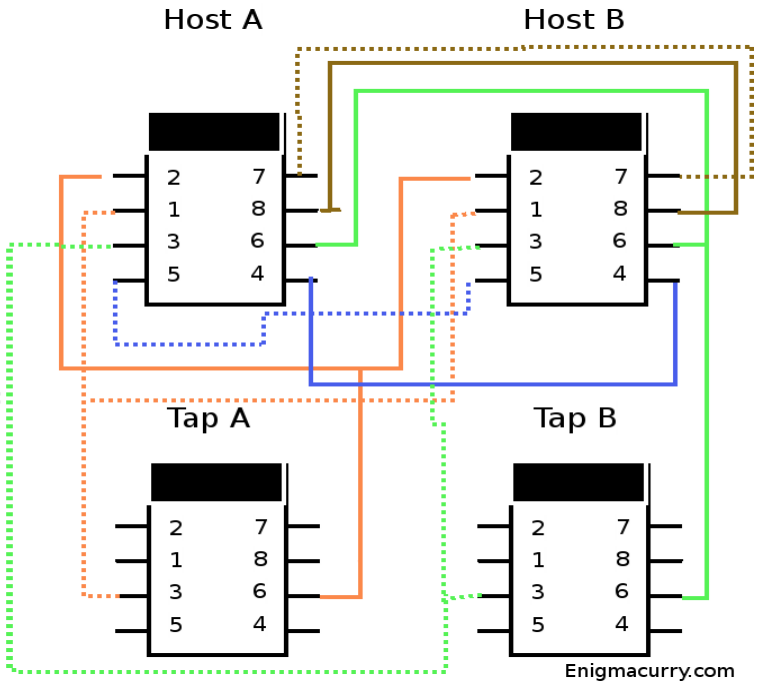
19“ Schrank mit Patch Kabel

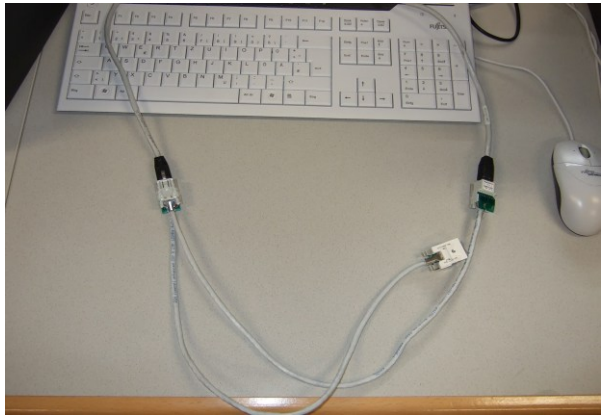


*(Tanenbaum
)*

Wiretapping

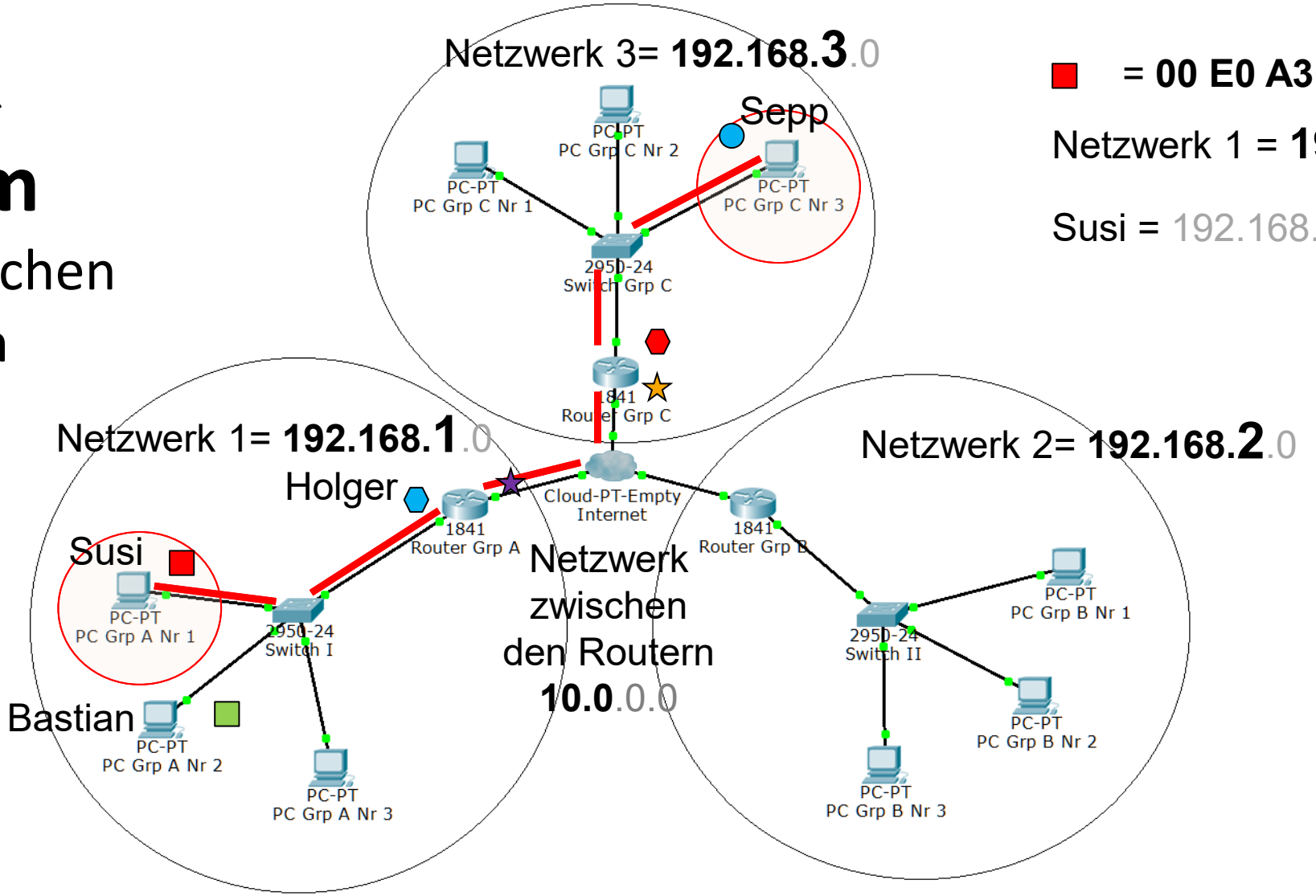






Netzwerk **Diagramm**

mit physikalischen und logischen Adressen und Weg der Nachricht von Susi zu Sepp.



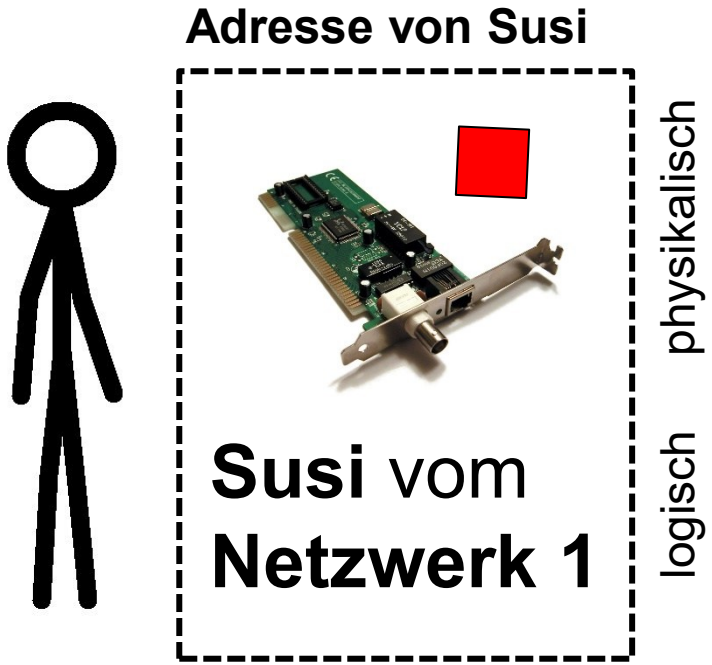
■ = **00 E0 A3 0D 24 25**

Netzwerk 1 = **192.168.1.0**

Susi = 192.168.1.**1**

Überblick

- **Netzwerkkomponenten haben eine eindeutige Identifikationen**
  Beispiel:  (Hardwareadresse*)
  Werden **vom Hersteller festgelegt**.
- **ben wir eine logische Adresse**
  Beispiel: "Netzwerk 1"
- **Netzwerkkomponenten vergeben wir im Netzwerk eine logische Adresse.** Beispiel: "Susi" in "Netzwerk 1"
- **Es gibt Router bzw. Gateways, die in andere Netzwerke führen**

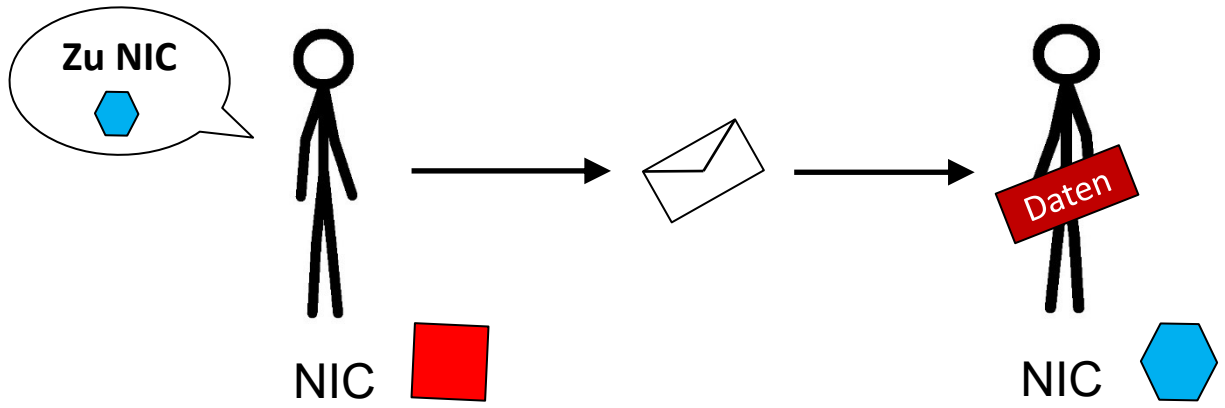


(* Media Access Control (MAC) Adresse: **40:B0:76**:A2:42:07$_{16}$
Die erste Hälfte der Adresse verweist auf den Hersteller (ASUSTek COMPUTER INC.)
Siehe *https://www.dein-ip-check.de/tools/macfinder*)

Kommunikation im physikalischen Netzwerk | Regeln

Netzwerkgeräte …

- … kommunizieren nur in Ihrem Netzwerksegment
- … adressieren sich über ihre Hardwareadresse (Symbole)
- … können mit einer Adresse FF:FF:FF:FF:FF:FF an alle anderen Geräte im Netzwerksegment eine Nachricht versenden*



Mit der **Hardwareadresse** werden Geräte erkannt und **physikalisch adressiert**

(Broadcast MAC Adresse)*

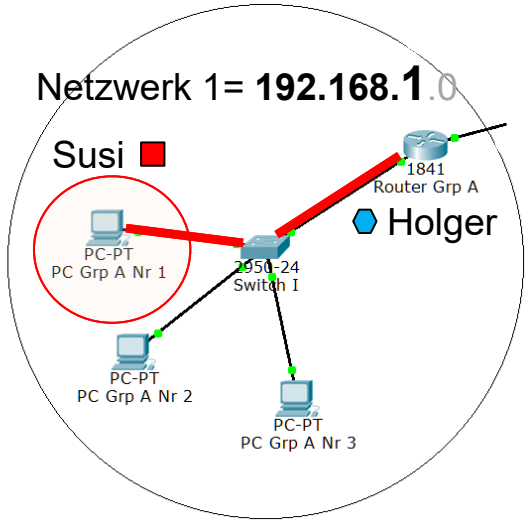
Kommunikation im logischen Netzwerk

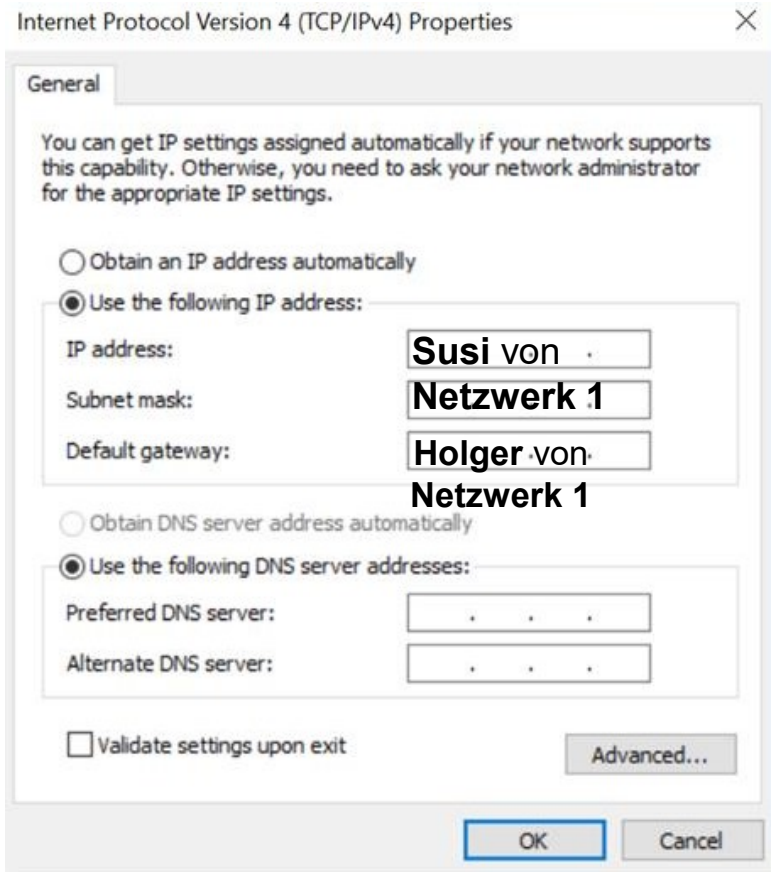
Kommunikationspartner in Netzen …

- … kommunizieren über die logische Adresse
- … sind in einem Netzwerk, welches in der logischen Adresse steckt und mit einer Subnetzmaske daraus berechnet werden kann

Wer ist mein Gateway?

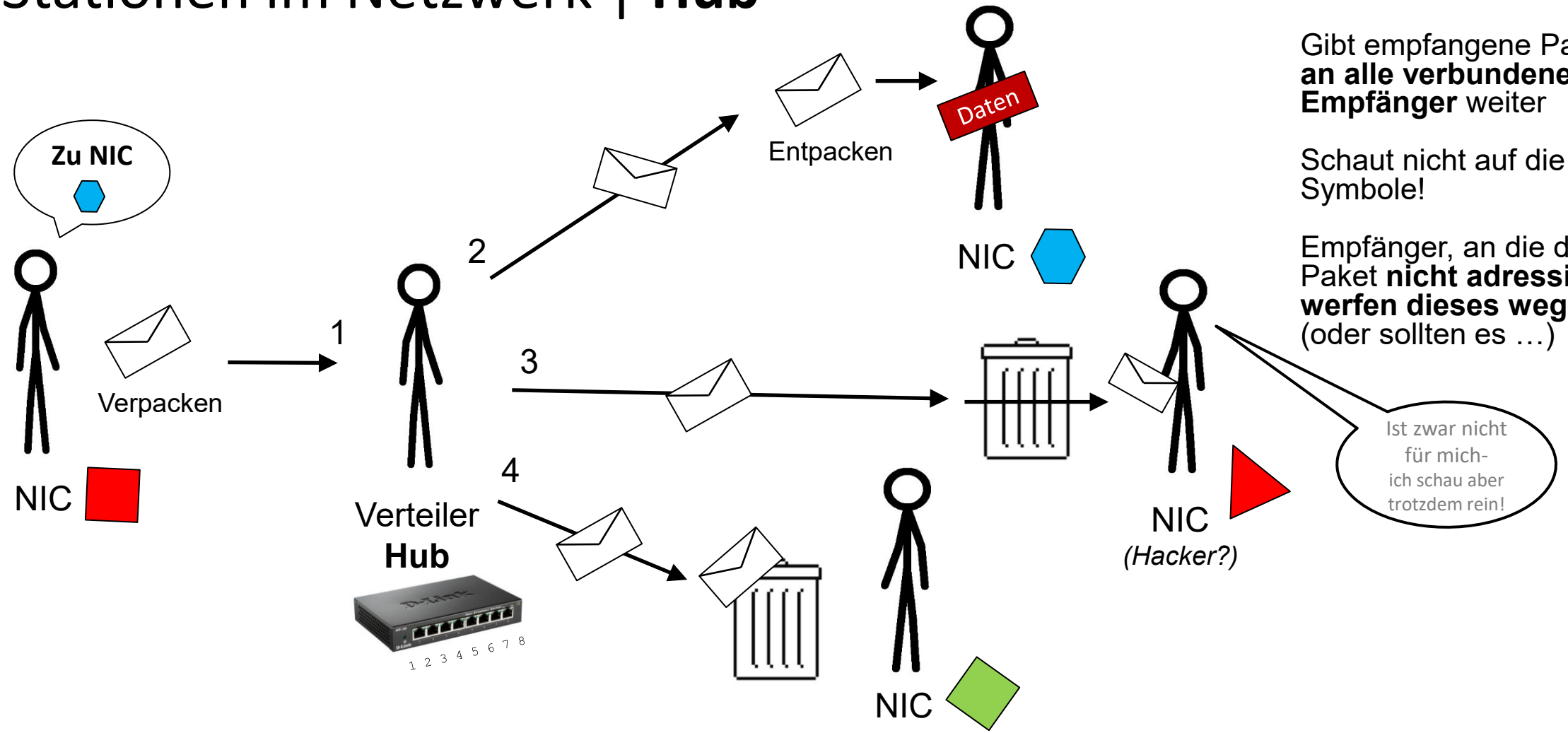
- Mein Weg nach draußen!
- Der "Standard-Gateway" ist der Route über den ich gehe, wenn ich in andere Netze will





Dialog unter Windows, in dem die **logische Adresse** (IP = Internet Protokoll) **und der Gateway** eingetragen werden

Stationen im Netzwerk | **Hub**

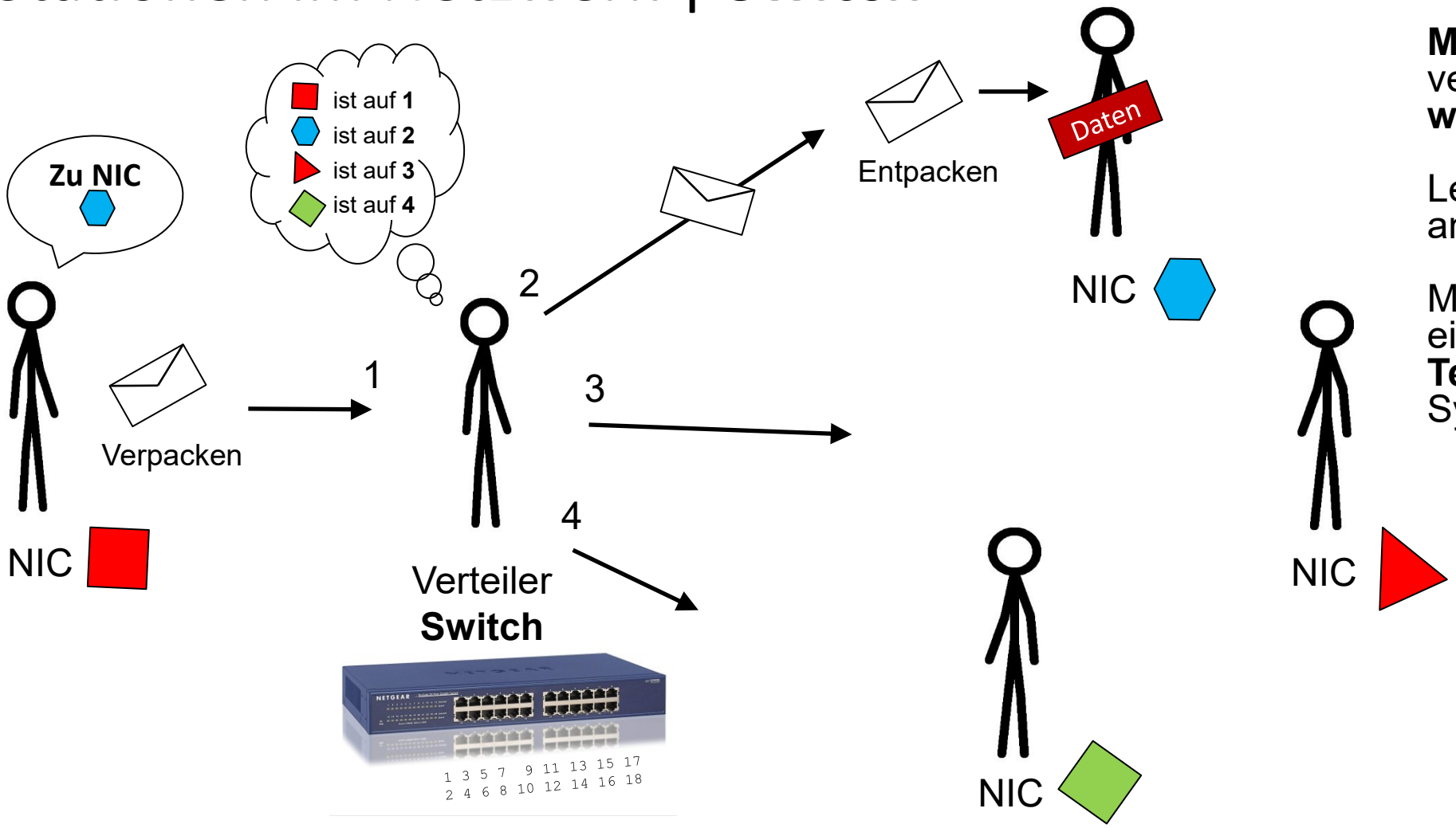


Gibt empfangene Pakete **an alle verbundenen Empfänger** weiter

Schaut nicht auf die Symbole!

Empfänger, an die das Paket **nicht adressiert** ist, **werfen dieses weg** (oder sollten es …)

Stationen im Netzwerk | **Switch**



Merkt sich, **welcher** verbundene **Empfänger** an **welchem Platz** ist

Leitet Pakete **zielgerichtet** an den Empfänger

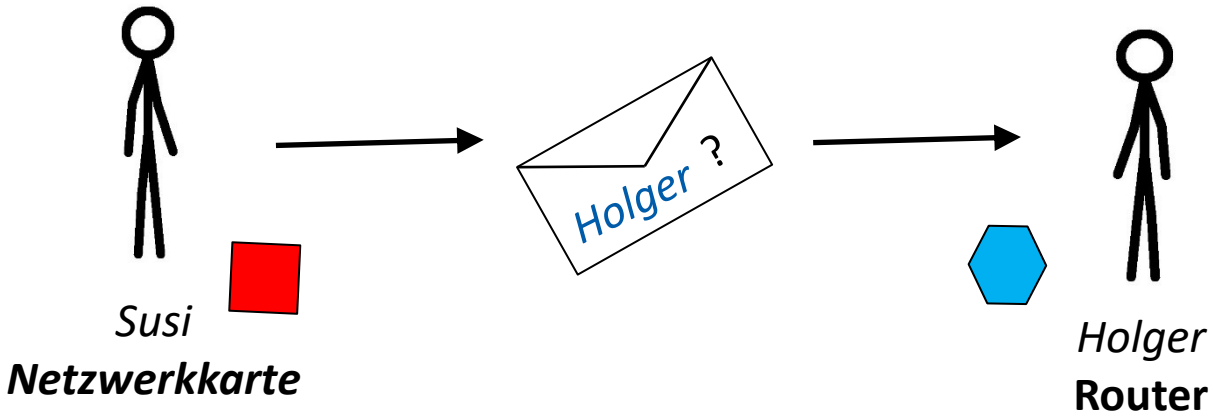
Muss sich zu Beginn aber eine **Tabelle aller Teilnehmer** aufbauen …
Symbol zu Steckplatz (Port)

Adressierung

Vereinfachtes Model

Kommunikation im Netzwerk - aber wie?

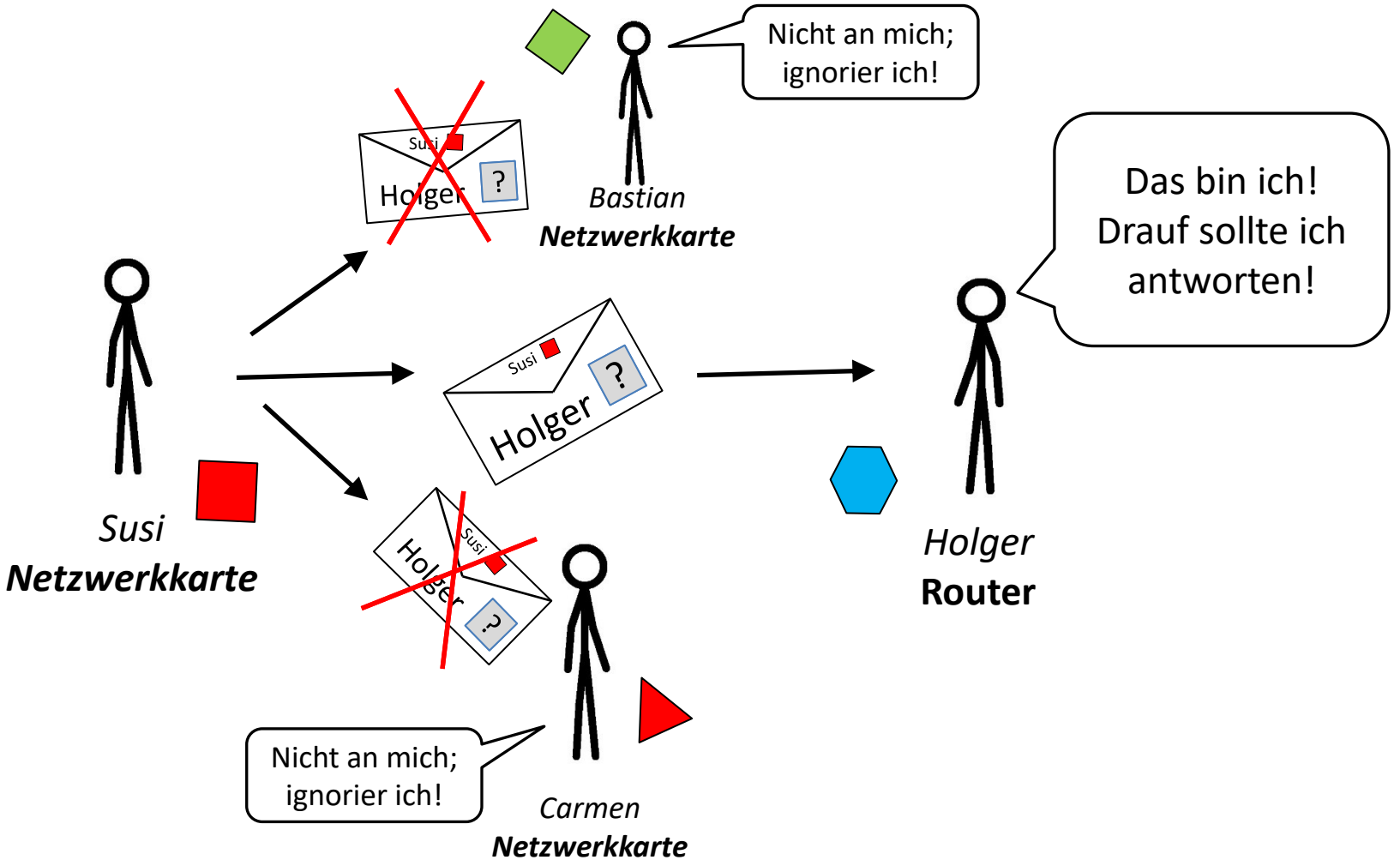
- Geräte adressieren sich über ihr Symbol
- Rollen adressieren sich über den Vornamen



Problem:

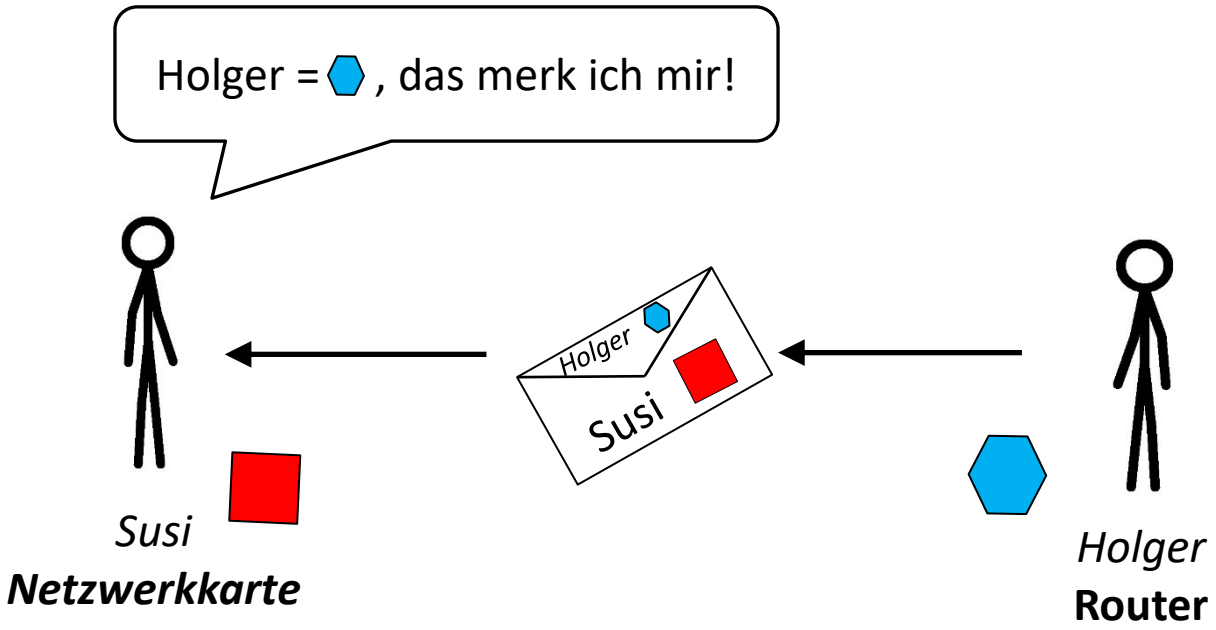
Wie kann man eine **Nachricht** an Holger senden, wenn man sein "Symbol" nicht kennt?

Address Resolution Protocol (ARP) | **Request**



(ARP Request)

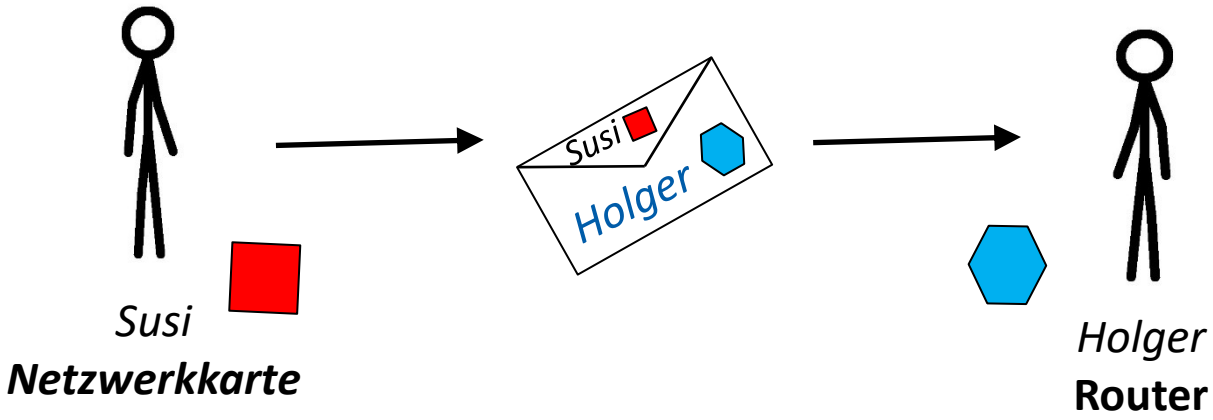
Address Resolution Protocol (ARP) | **Response**



(ARP Response)

Kommunikation im Netzwerk – nach ARP

- Geräte adressieren sich über ihr Symbol
- Rollen adressieren sich über den Vornamen

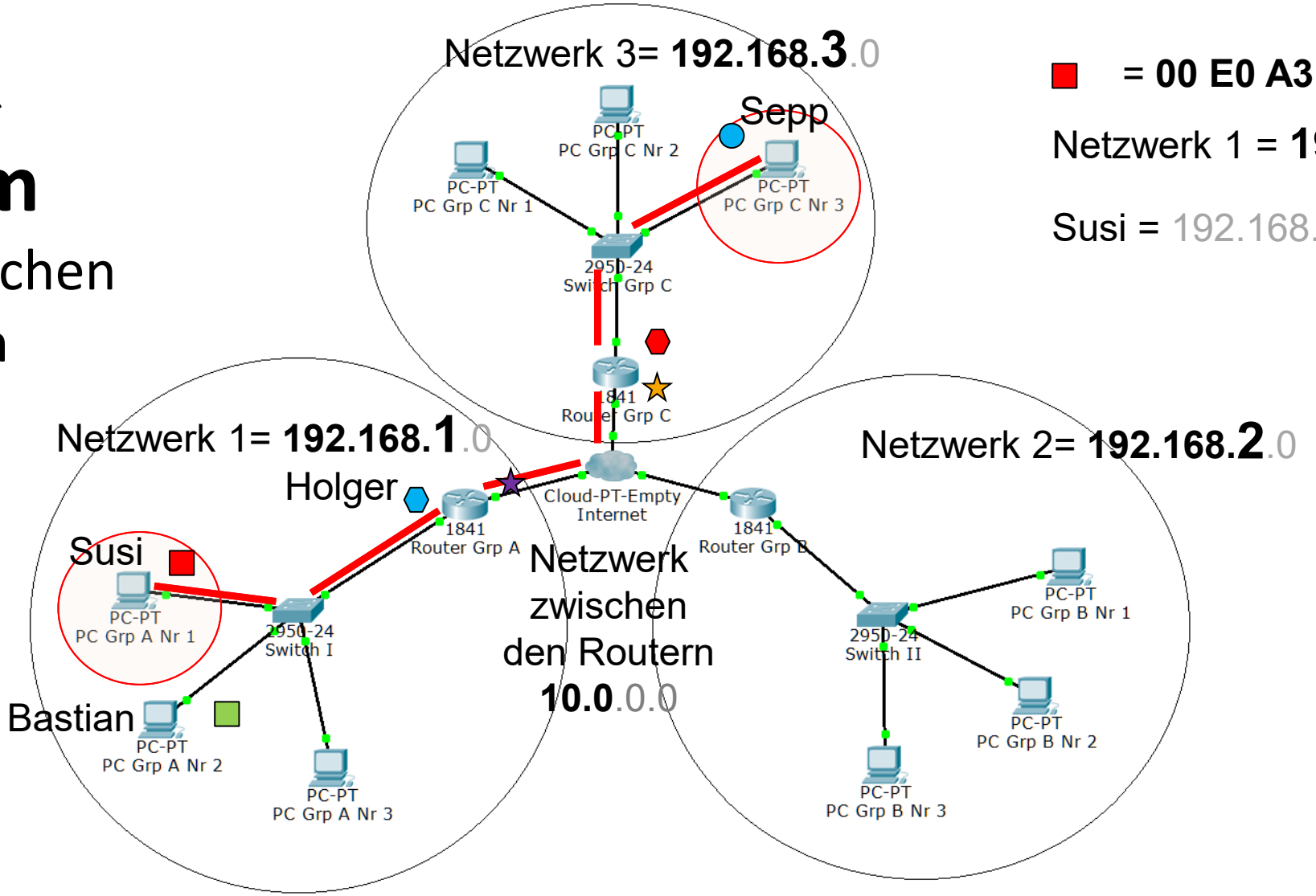


Lösung:

Man **fragt mal in die Runde**, wer **Holger** ist und **welches Symbol** er hat! Das **merkt** man sich!

Netzwerk **Diagramm**

mit physikalischen und logischen Adressen und Weg der Nachricht von Susi zu Sepp.



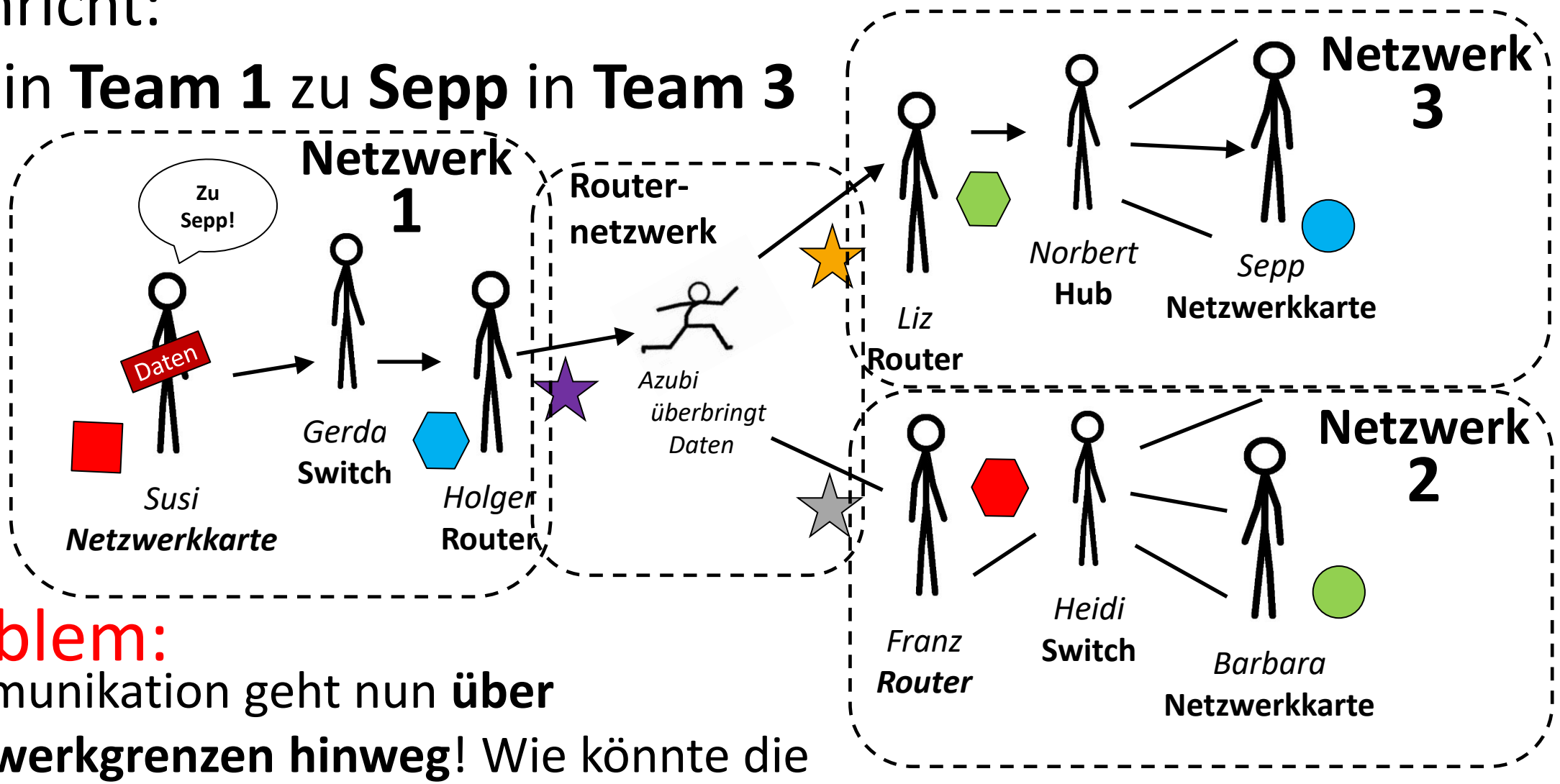
= **00 E0 A3 0D 24 25**

Netzwerk 1 = **192.168.1.0**

Susi = 192.168.1.**1**

Nachricht:

Susi in **Team 1** zu **Sepp** in **Team 3**

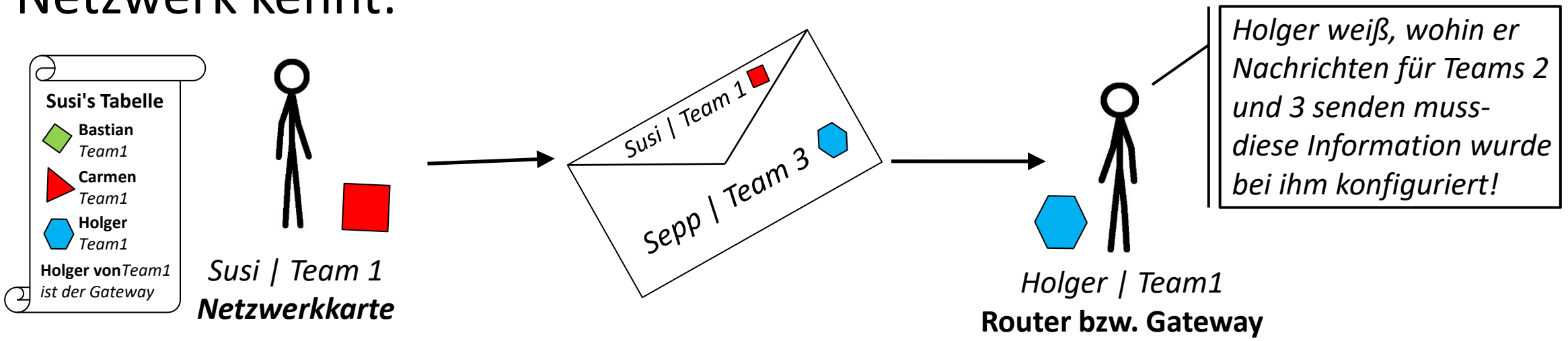


Problem:

Kommunikation geht nun **über Netzwerkgrenzen hinweg**! Wie könnte die Kommunikation jetzt aussehen?

Lösung: **Nachricht an den Gateway**

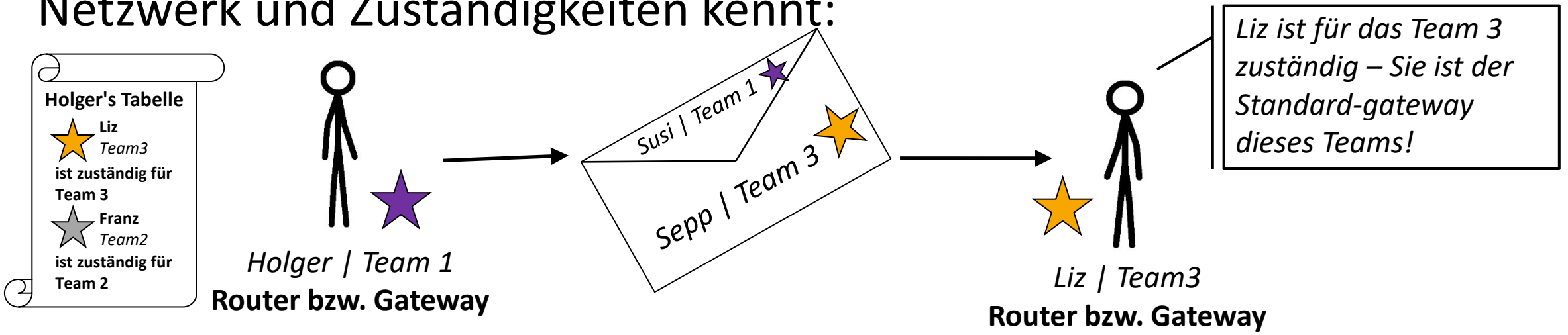
- Wir gehen davon aus, dass Susi alle Teilnehmer in ihrem Netzwerk kennt:



- **Schritt 1**: Susi sendet eine Nachricht **logisch an Sepp von Team 3 physikalisch an den Gateway**

Lösung: **Nachricht von Gateway an Gateway**

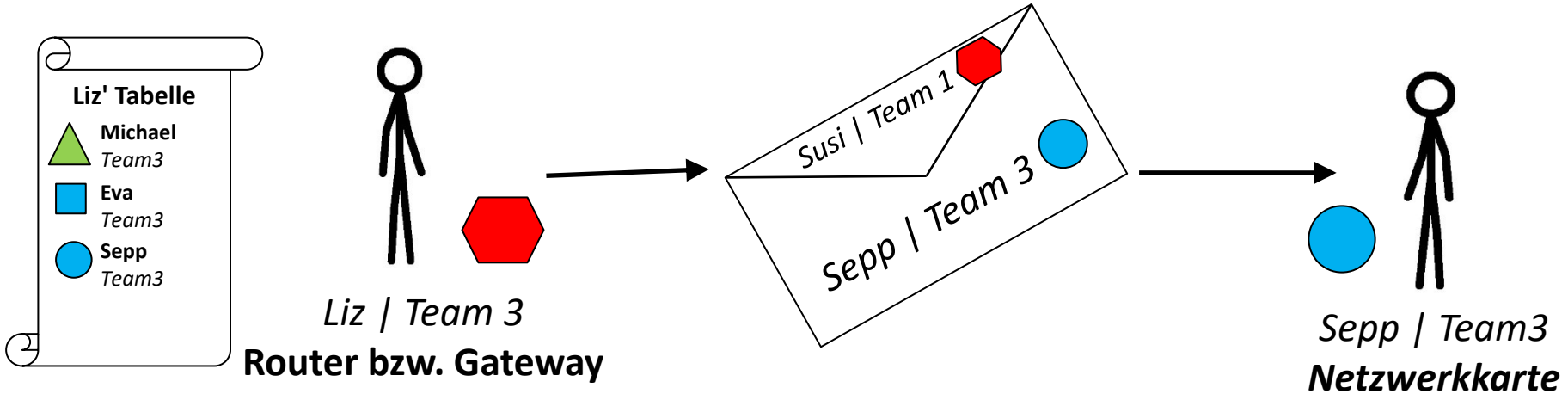
- Wir gehen davon aus, dass Holger alle Teilnehmer in seinem Router-Netzwerk und Zuständigkeiten kennt:



- **Schritt 2**: Holger sendet eine Nachricht **logisch an Sepp von Team 3 physikalisch an den Gateway von Team 3**

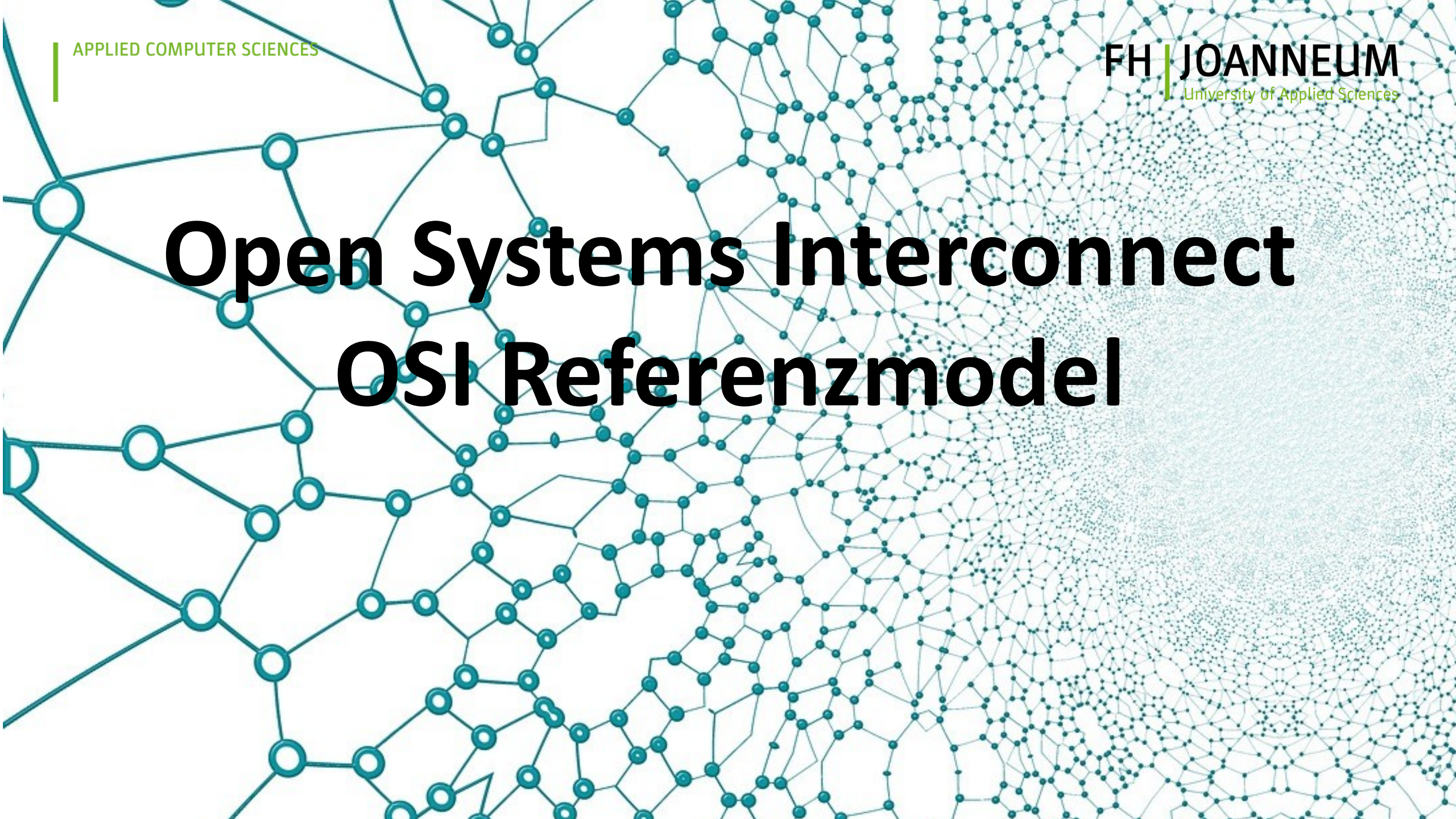
Lösung: **Nachricht von Gateway an Empfänger**

- Wir gehen davon aus, dass Liz alle Teilnehmer in ihrem Netzwerk kennt:



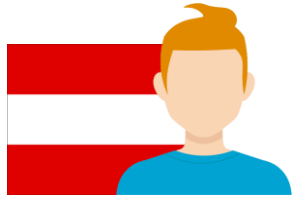
- **Schritt 3**: Liz sendet eine Nachricht **logisch und physikalisch an Sepp von Team 3**

FH | JOANNEUM
University of Applied Sciences



Open Systems Interconnect
OSI Referenzmodel

Beispiel – Telefonanruf nach Spanien







Teilnehmer in Österreich

Virtuelle Verbindung
Protokoll

Teilnehmer in Spanien

Schnittstelle

Übersetzer

Virtuelle Verbindung
Protokoll

Übersetzer

Schnittstelle

Telefon

Telefon

Schnittstelle

Vermittlung

Vermittlung

Schnittstelle

Kabel

Signale
Physische Verbindung

Kabel

Begriffe

- **Schnittstelle**
  - Übergang zwischen 2 übereinanderliegenden Schichten
  - Regeln der Kommunikation über die Schnittstellen bilden die Schnittstellendefinition
- **Dienste**
  - Leistungen, die eine Schicht der darüber liegenden zur Verfügung stellt
- **Protokolle**
  - Kommunikationsvorschrift von Partnern der gleichen Schicht.
  - Format der ausgetauschten Daten
- **Virtuelle Verbindung**
  - Kommunikation zwischen 2 Partnern der gleichen Schicht

OSI – Open Systems Interconnect

Kommunikation in **sieben Schichten** mit diesen Zielen:

- Eine **Schicht** soll dort erstellt werden, wo ein neuer **Abstraktionsgrad** nötig ist.
- Jede Schicht soll eine **genaue definierte Funktion** erfüllen.
- Bei Funktionswahl sollen international **genormte** Protokolle berücksichtigt sein.
- Die Grenzen zwischen den Schichten sollen so sein, dass der **Informationsfluss** über sie **möglichst gering** bleibt.
- Die Anzahl der Schichten soll eine **handliche Architektur** mit unterschiedlichen Funktionen in unterschiedlichen Schichten sein.

Die glorreichen Sieben

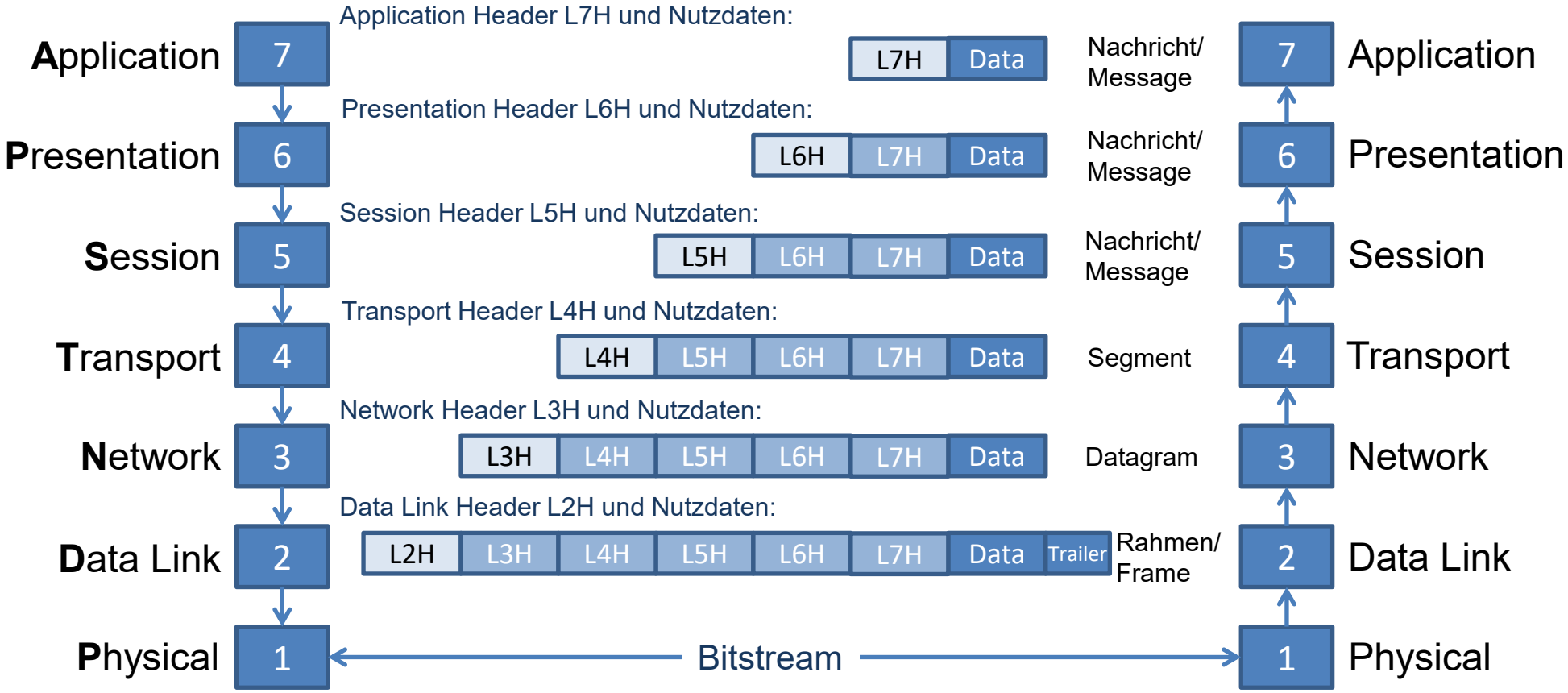
OSI Datenkapselung und -transport

Anwendungsorientiert:

- Anwendungsschicht
- Präsentationsschicht
- Sitzungsschicht

Transportorientiert:

- Transportschicht
- Vermittlungsschicht
- Sicherungsschicht
- Physische Schicht

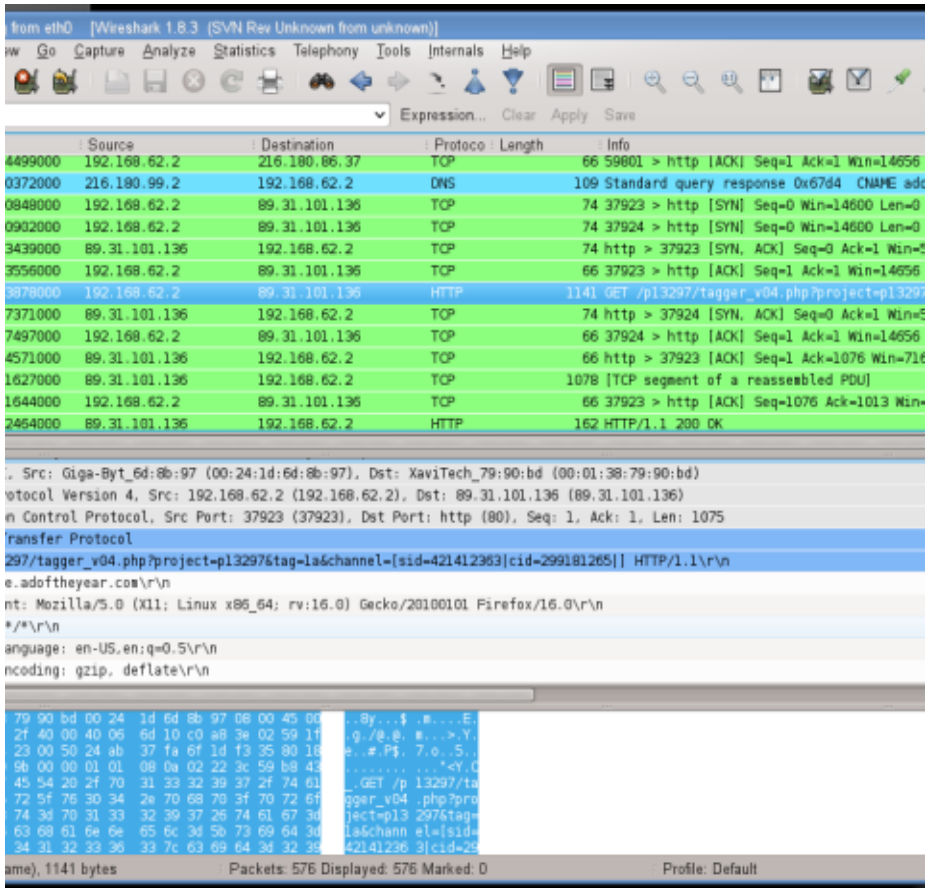


„**P**lease **D**o **N**ot **T**hrow **S**alami **P**izza **A**way“

Werkzeuge | **Wireshark**

- **Analysiert** und **visualisiert Datenprotokolle** auf
- Bietet Möglichkeiten zum **Filtern, Aufbereiten und der Visualisierung**
- **Site:** https://www.wireshark.org/
- **Filterreferenz:** https://www.wireshark.org/docs/dfref/

- **Demo**



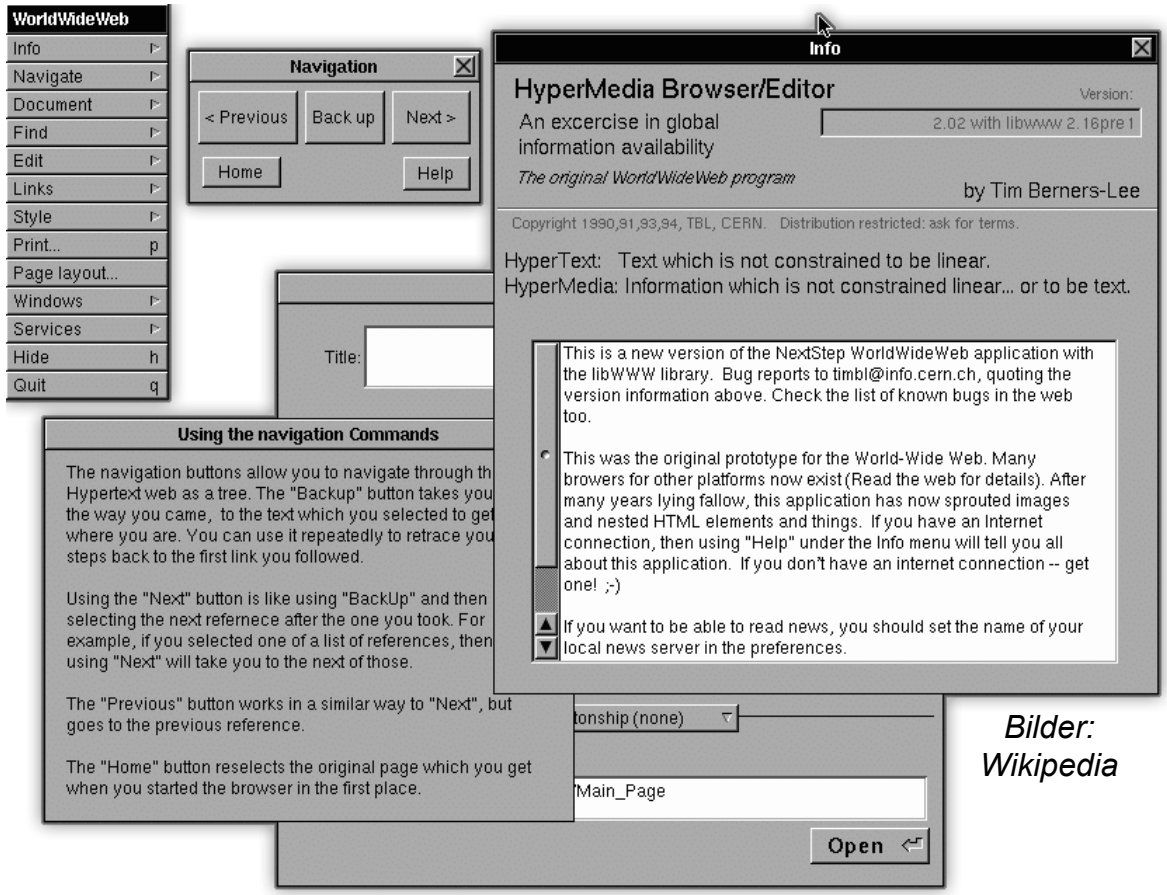
Fallbeispiel: **Im Internet browsen**

Historisches Logo des WWW von Robert Cailliau

*Erster Web**server***

*Erstes **Bild** über das WWW*

- Das **WWW – World Wide Web** ist einer der wichtigsten Dienste, der das Internet erst populär gemacht hat
- Es wurde 1988 bis 1991 von **Tim Berners-Lee** (Cern) entwickelt
- Das WWW ist **NICHT** das Internet !!

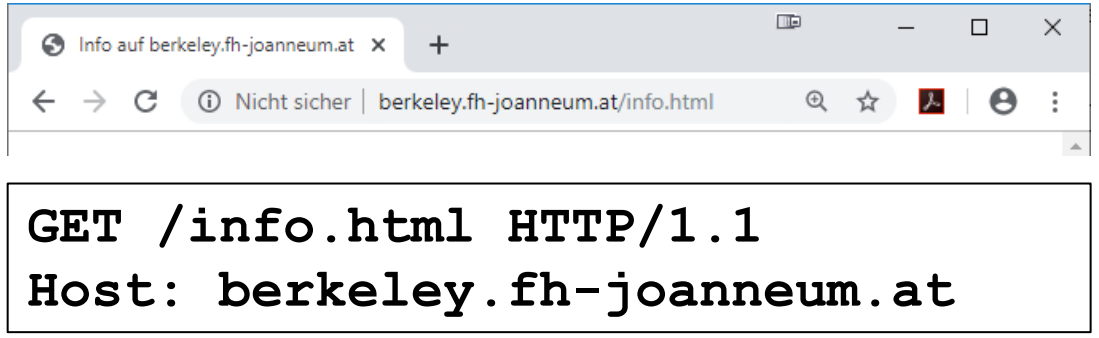


Bilder: Wikipedia

*Erster Web**browser***

Fallbeispiel: **Client - Server Kommunikation**

- **Client**
  Web Browser
  z.B. Google Chrome)




```
GET /info.html HTTP/1.1
Host: berkeley.fh-joanneum.at
```

- **sendet**
  über das Netzwerk eine
  HTTP*- **Anfrage**

- an den
  Web-**Server**
  z.B. Apache 2

*(*HTTP = Hypertext Transfer Protocol)*

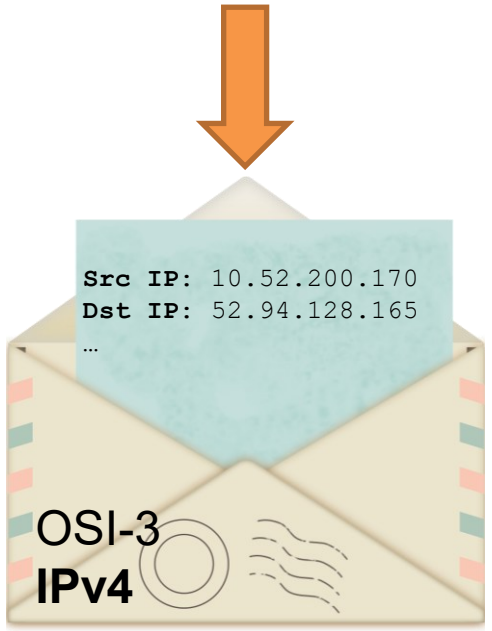
OSI Kapselung

Beispiel HTTP

Port 80

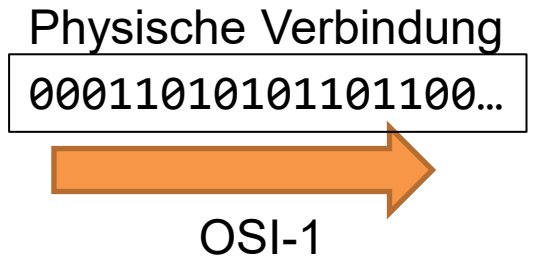
Client Request

Ein Paket vom ***Client*** *geht an den* ***Server***



Legende:

Src …	Source (Quelle)
Dst …	Destination (Ziel)
IP …	Internet Protokoll Logische Adressierung
MAC …	Media Access Control Hardware Adresse Zugriff auf das Medium Netzwerkkarte
Port …	"Nummer" des Dienstes, der am Server angesprochen wird- bzw. Nummer für "Rück-Antwort" von diesem Dienst



Fallbeispiel: **Client - Server Kommunikation (2)**

von Client

von
Hardware Adresse
Client PC

an
Hardware Adresse
Gateway

No.	Time	Source	Destination	Protocol	Length	Info
3382	22.991350	10.52.200.132	91.229.57.56	HTTP	698	GET /info.html HTTP/1.1

```
Frame 3382: 698 bytes on wire (5584 bits), 698 bytes captured (5584 bits) on interface 0
Ethernet II, Src: 40:b0:76:a2:42:07, Dst: 00:08:e3:ff:fd:2c
Internet Protocol Version 4, Src: 10.52.200.132, Dst: 91.229.57.56
Transmission Control Protocol, Src Port: 53638, Dst Port: 80, Seq: 1, Ack: 1, Len: 644
Hypertext Transfer Protocol
  GET /info.html HTTP/1.1\r\n
  Host: berkeley.fh-joanneum.at\r\n
  Connection: keep-alive\r\n
  Cache-Control: max-age=0\r\n
  Upgrade-Insecure-Requests: 1\r\n
  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/78.0.3904.97 Safari/537.36\r\n
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3\r\n
  Accept-Encoding: gzip, deflate\r\n
  Accept-Language: de-DE,de;q=0.9,en-US;q=0.8,en;q=0.7\r\n
  Cookie: _ga=GA1.2.1115739603.1574071763; _gid=GA1.2.764581842.1574071763\r\n
  If-None-Match: "9b-5979e1fe538ec-gzip"\r\n
  If-Modified-Since: Mon, 18 Nov 2019 12:31:19 GMT\r\n
```

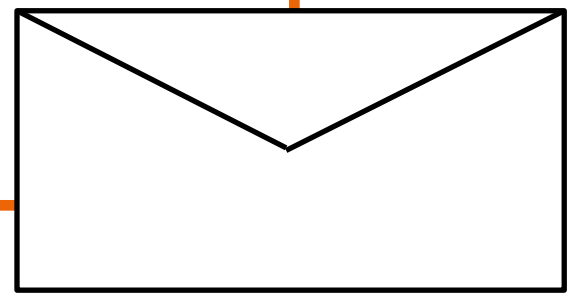
Port für
HTTP

von
Logische Adresse
Client PC

an
Logische Adresse
Webserver

Aus "ipconfig /all"
Netzwerkkarte von Client:
Physische Adresse : 40-B0-76-A2-42-07
IPv4-Adresse . . . . : 10.52.200.132
Standardgateway : **10.52.1.254**

Aus "arp –a":
Schnittstelle: 10.52.200.132
Internetadresse Physische Adresse **10.52.1.254** **00-08-e3-ff-fd-2c**

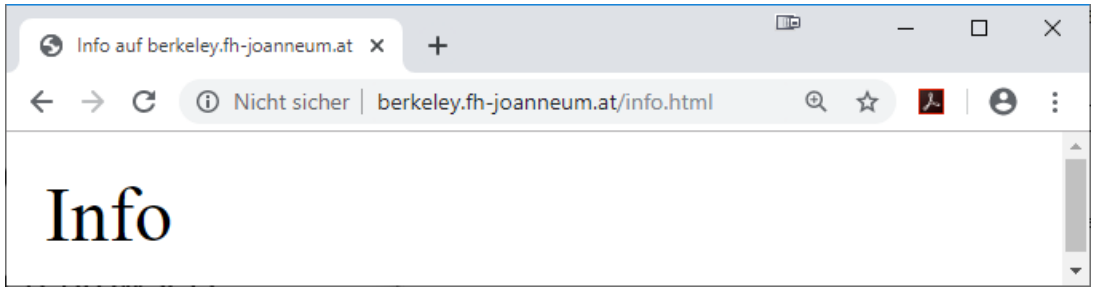


Fallbeispiel: **Server – Client Kommunikation (1)**

- Der **Server** sendet eine (HTTP-) **Antwort** an den **Client** (=Browser) zurück

```
HTTP/1.1 200 OK
Content-Type: text/html

<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Info auf berkeley.fh-joanneum.at</title>
  </head>
  <body>
        Info
  </body>
</html>
```

- und deren Inhalt zeigt ein Browser als (HTML) **Dokument** gerendert an



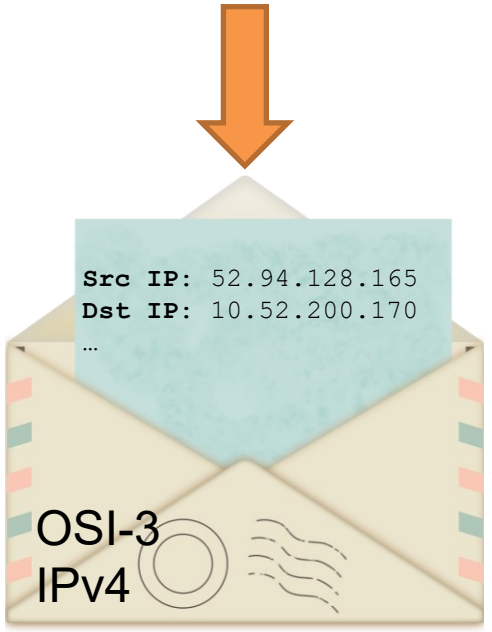
OSI Kapselung

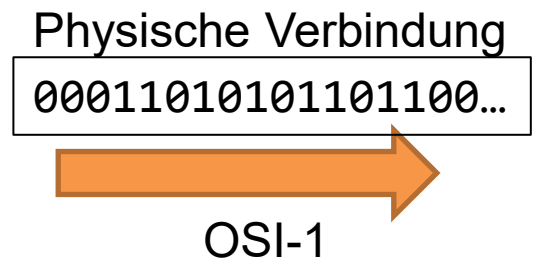
Beispiel HTTP

Port 80

Server Response

Ein Paket vom ***Server*** *geht an den* ***Client*** *zurück*





Legende:

Src …	Source (Quelle)
Dst …	Destination (Ziel)
IP …	Internet Protokoll Logische Adressierung
MAC …	Media Access Control Hardware Adresse Zugriff auf das Medium Netzwerkkarte
Port …	"Nummer" des Dienstes, der am Server angesprochen wird- bzw. Nummer für "Rück-Antwort" von diesem Dienst

Fallbeispiel: Server – Client Kommunikation (1)

von Server

von
Hardware Adresse
Gateway

an
Hardware Adresse
Client PC

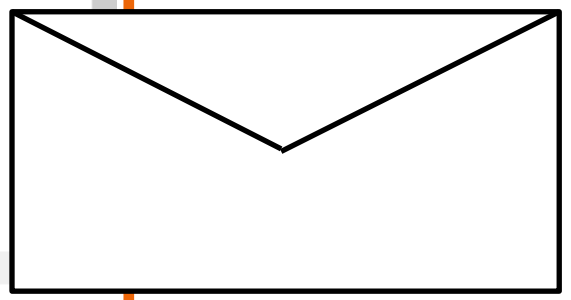
No.	Time	Source	Destination	Protocol	Length	Info
3483	13.250701	91.229.57.56	10.52.200.132	HTTP	559	HTTP/1.1 200 OK (text/html)

```
Frame 3483: 559 bytes on wire (4472 bits), 559 bytes captured (4472 bits) on interface 0
Ethernet II, Src: 00:08:e3:ff:fd:2c, Dst: 40:b0:76:a2:42:07
Internet Protocol Version 4, Src: 91.229.57.56, Dst: 10.52.200.132
Transmission Control Protocol, Src Port: 80, Dst Port: 54074, Seq: 1, Ack: 619, Len: 505
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Mon, 18 Nov 2019 13:20:42 GMT\r\n
  Server: Apache/2.4.25 (Debian)\r\n
  Last-Modified: Mon, 18 Nov 2019 12:31:19 GMT\r\n
  ETag: "9b-5979e1fe538ec-gzip"\r\n
  Accept-Ranges: bytes\r\n
  Vary: Accept-Encoding\r\n
  Content-Encoding: gzip\r\n
  Access-Control-Allow-Origin: *\r\n
  Content-Length: 137\r\n
  Keep-Alive: timeout=5, max=100\r\n
  Connection: Keep-Alive\r\n
  Content-Type: text/html\r\n
  \r\n
  [HTTP response 1/1]
  [Time since request: 0.004144000 seconds]
  [Request in frame: 3481]
  [Request URI: http://berkeley.fh-joanneum.at/info.html]
  Content-encoded entity body (gzip): 137 bytes -> 155 bytes
  File Data: 155 bytes
Line-based text data: text/html (11 lines)
```

Port für
HTTP

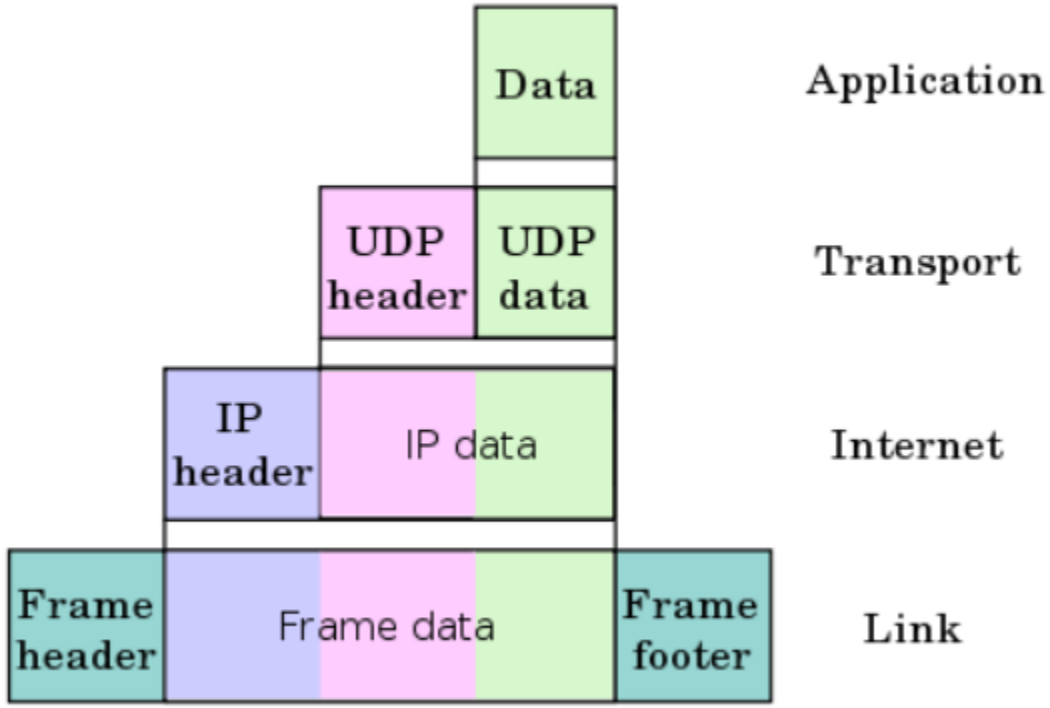
von
Logische Adresse
Webserver

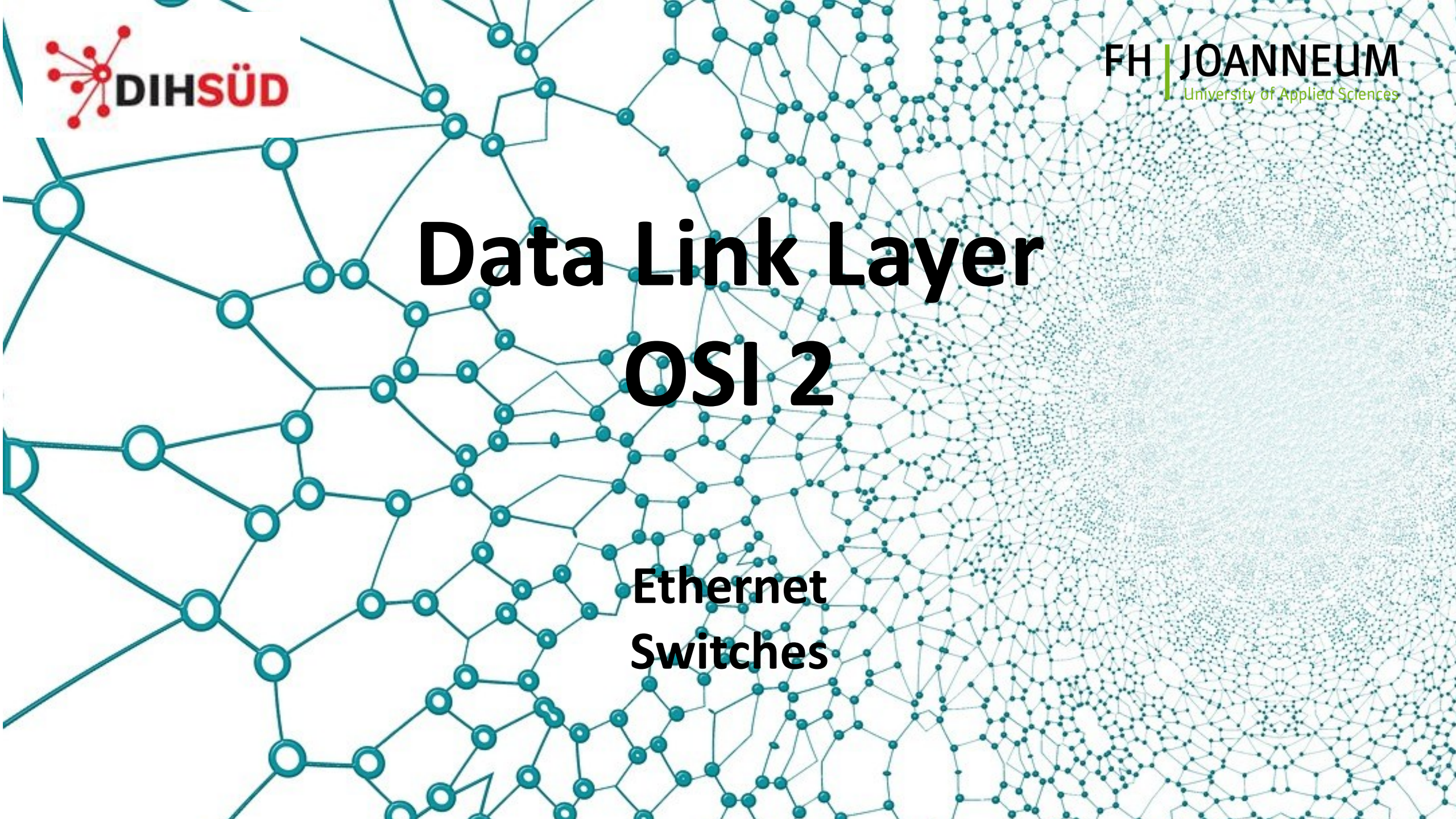
an
Logische Adresse
Client PC



Internet Protocol Suite

Schicht	Protokolle
Application	FTP, Telnet, SMTP, …
Transport	TCP, UDP
Network	IP, ICMP, IGMP,…
Link	Ethernet





DIHSÜD

FH JOANNEUM
University of Applied Sciences

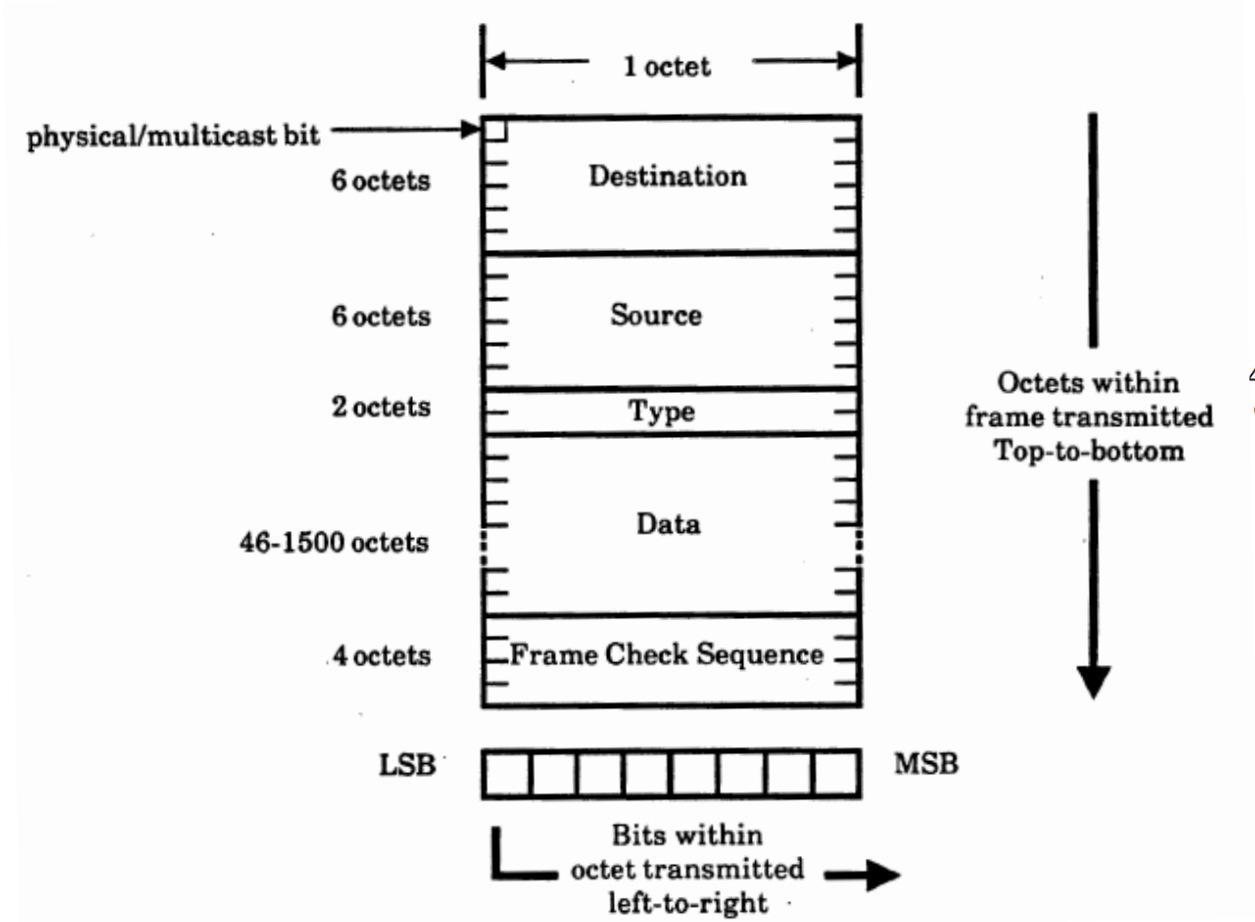
Data Link Layer
OSI 2

Ethernet
Switches

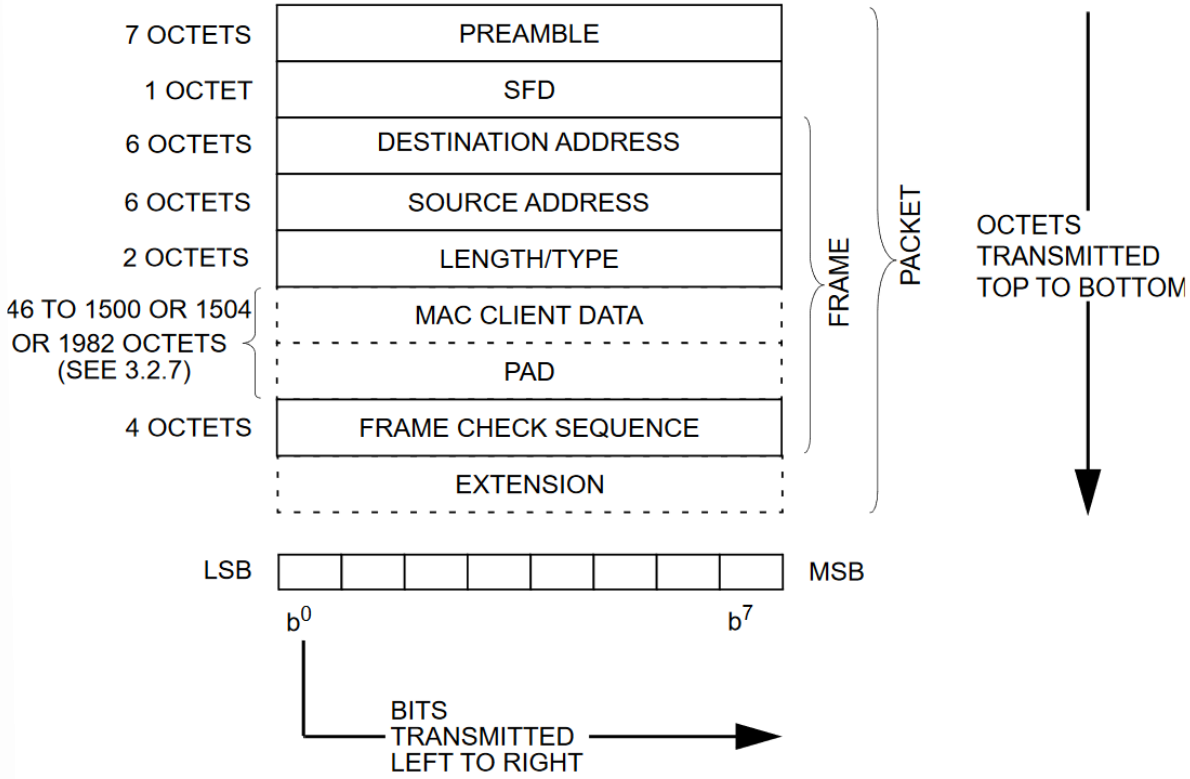
Ethernet

- **Spezifikation** von Funktionalitäten in
  - **physischer Schicht** (Layer 1)
  - **Sicherungsschicht** (Layer 2)
- Packt das **IP Datagramm** in einen **Ethernet Frame (Rahmen).**
- Fügt die **physikalischen Adressen** der Quelle (MUA) und des Ziels (MTA von Jane) dazu. (**MAC - Media Access Control Adressen**)
- Versendet die **Frames über das Netzwerkinterface**.
- "**Ether**"(dt. Äther) , weil dieses Protokoll **nicht auf ein Übertragungsmedium beschränkt** ist: Medien wie Kabel, Luftschnittstelle, Lichtwellen, usw. sind möglich
- Arbeitet innerhalb einer **Kollissionsdomäne**
- Ursprünglich von DEC, Intel und Xerox (DIX) spezifiziert, dann auch in IEEE802.3

Ethernet | **Ethernet II (DIX)** und **Ethernet IEEE 802.3**

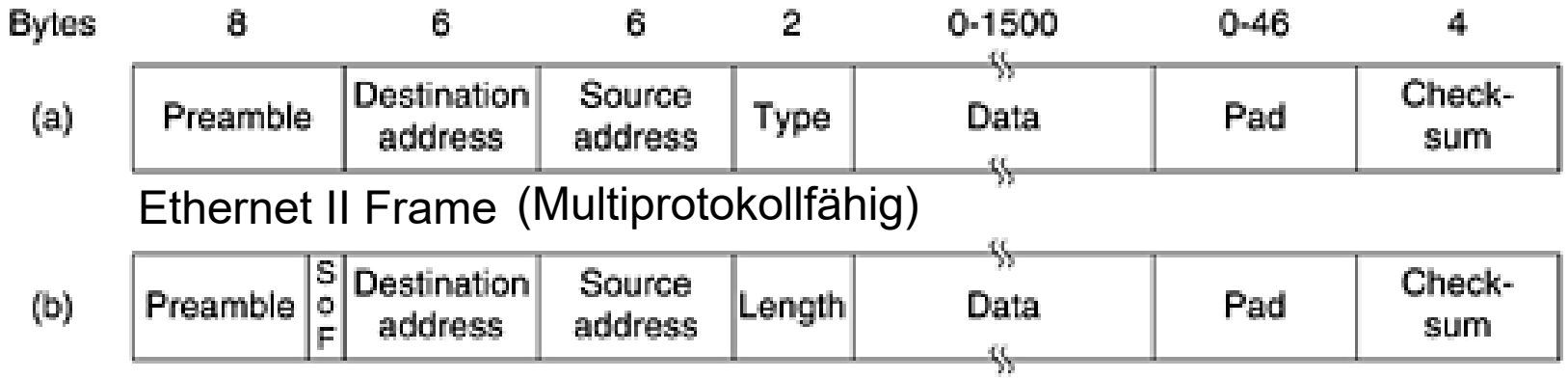


Ethernet II Frame (Ethernet Version 2.0, DIX, 1982)
Achtung: Hat auch eine führende 7B Preambel!



Ethernet 802.3 Frame (Ethernet IEEE802.3, 2018)

Ethernet | **Media Access Control** (MAC) Teilschicht



Ethernet II Frame (Multiprotokollfähig)

802.3 „Raw“ Frame (Novell, nicht Multiprot.-fähig)

vgl. Computer Networks, Tanenbaum, 5th Edition, 2011

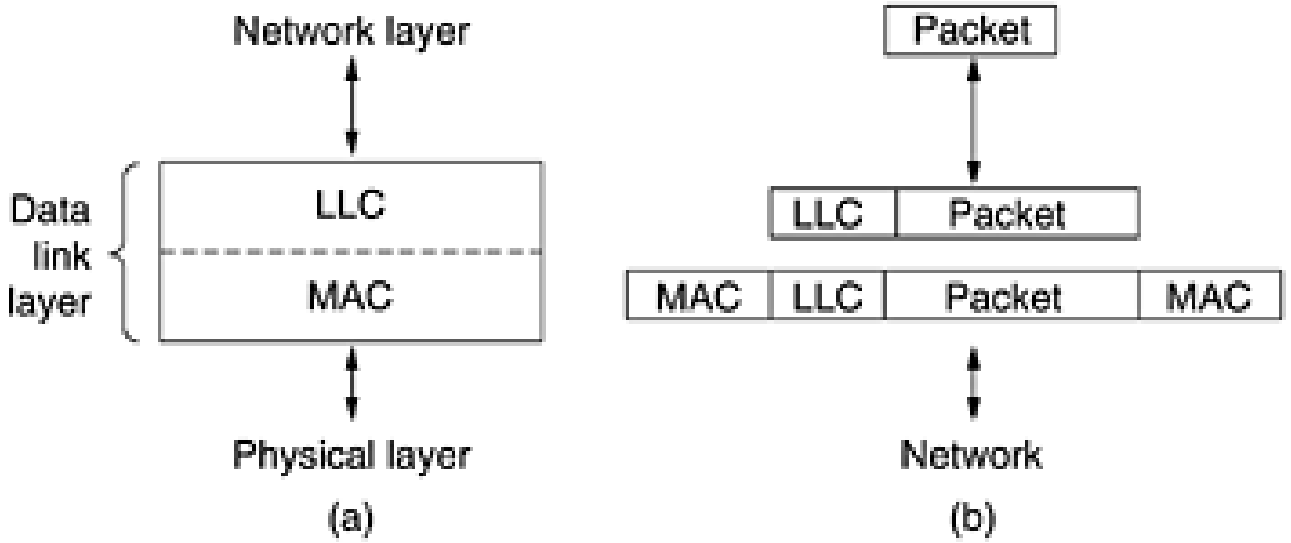
- **Präambel** zur Taktsynchronisierung: 7-Byte Folge mit 1010101
- **Start of Frame** (SoF): 1 Byte
- **Destination Address** (Zieladresse)(6 Byte)
- **Source Address** (Quelladresse)(6 Byte)
- **Length**: Länge des Datenfeldes
- **Checksum**: CRC Prüfsumme
- **Type**: Spezifikation des verwendeten Protokolls im Datenbereich

Collision Domain (Kollisionsdomäne)

- Wenn Netzwerkgeräte um ein **gemeinsames Übertragungsmedium konkurrieren** spricht man von einer Kollisionsdomäne
- Teil des Ethernet II Standards
  - Bei **Voll-Duplex** geschehen **keine Kollisionen** mehr
  - Kollisionen zwischen **Netzwerkkarte und Switch Port bei Halb-Duplex**
  - Aus dem folgenden **10GB Ethernet Standard herausgefallen**
- **Wichtig bei WLAN**, weil hier das **gleiche Medium (Funk)** verwendet wird.

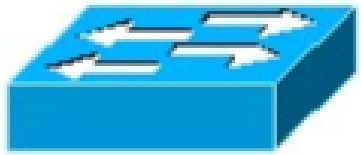
Ethernet | **Logical Link Layer** (LLC) Teilschicht

- Folge- und
- Bestätigungsnummer
- Unzuverlässig, bestätigter oder zuverlässiger verbindungsorientierte Verbindung

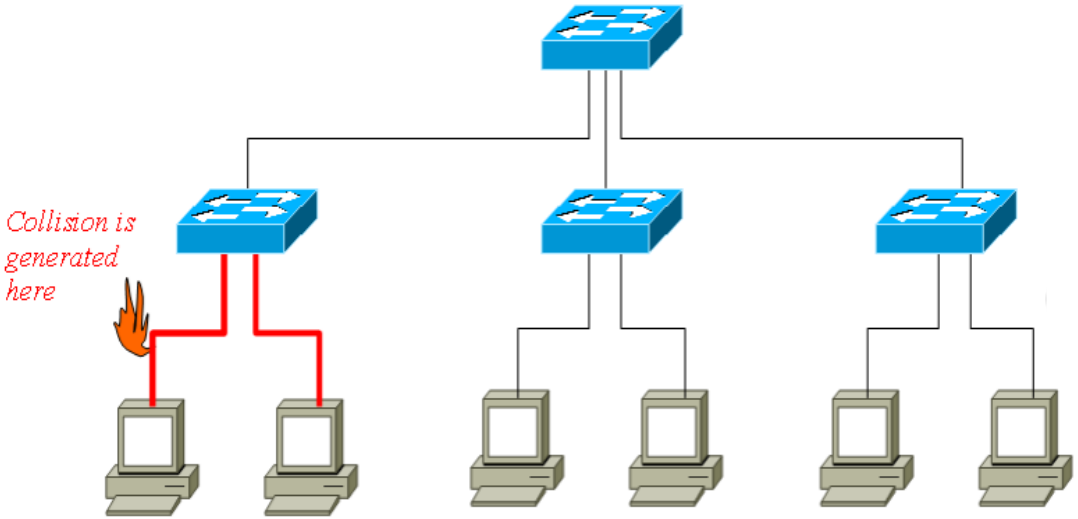


vgl. Computer Networks, Tanenbaum, 5th Edition, 2011

Switch (Schicht 2 Netzwerkverteiler)



Symbol
(Cisco)



(Cisco)

Kollisionen kommen nur zwischen Netzwerkgerät und Switchport **im Half-Duplex Modus** zustande

Frame Forwarding (Weiterleitung der Rahmen)

- Am Switch existiert eine **MAC Adress-Tabelle, anhand der er Nachrichten zielgerichtet weiterleitet**
- Die Tabelle wird **nach Einschalten des Switchs** anhand der Nachrichten **aufgebaut**!
- Wenn der Switch **keine Adresszuordnung** hat, geht er in den „**Flooding**“ **Modus** und sendet das Paket **wie ein Hub an alle**
- Wenn das **Ziel am selben Port** liegt, wird **nicht weitergeleitet** („**Filtering**“)
- Wenn die **Adresstabelle überläuft**, schaltet der Switch in den **„Flooding“ Modus**.

Erweiterte Funktionen von "managed" Switches"

- **Ports ein-** und **aus**schalten
- **Verbindungsgeschwindigkeit** und Duplex Modi einstellen
- **Prioritäten** für Ports festlegen
- **MAC Adressen filtern**
- Das **Spanning Tree Protocol (STP)**
- Überwachen des Geräts und der Ports über das **Simple Network Management Protokoll (SNMP)**
- **Port Spiegelung**
- **Link Aggregation** („bonding“/“trunking“)
- **Virtuelle LANs einsetzen (VLANs)**

Switches | **Attacken**

MAC Flooding

- Überladen der MAC Adresstabelle
- Senden von Paketen an das gesamte Netzwerk
- Daten könne mitge-"sniffed" werden

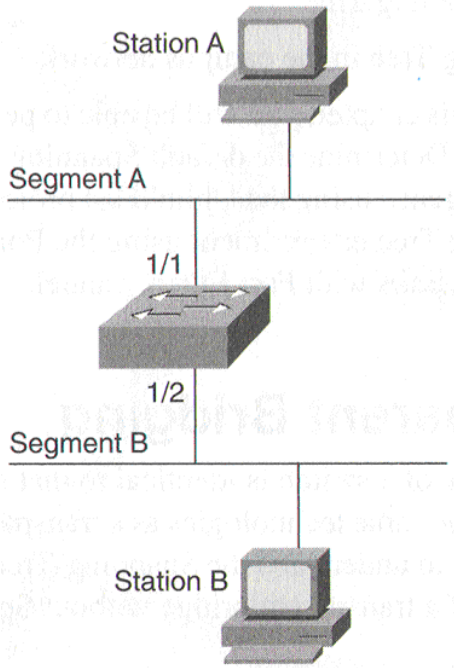
MAC Spoofing

- Angreifer manipulieren MAC Addressen für z.B. Router
- Ziele: Denial of Service, Man in the Middle

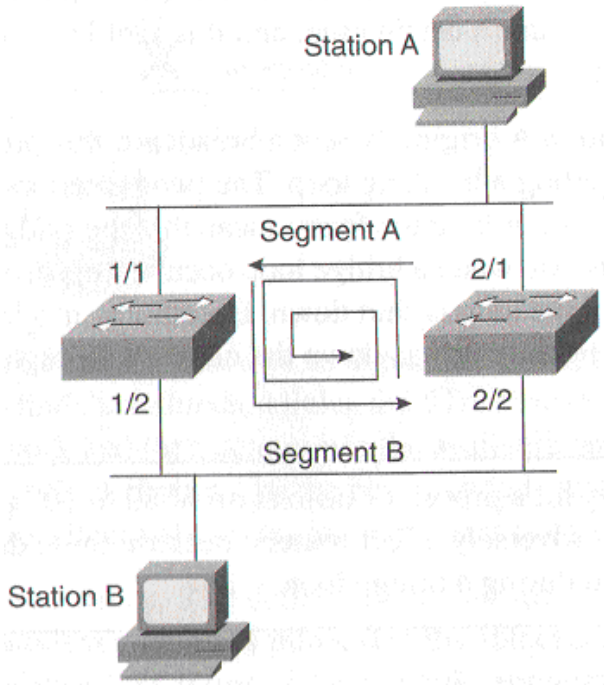
Gegenmaßnahmen (Port Security)

- Limitieren der MAC Adressen pro Switch Port
- Switch-port für bestimmte Zeit oder für immer abschalten
- Benachrichtigung / Warnung an den Administrator
- Kontrolle der Quellen

„Bridging“



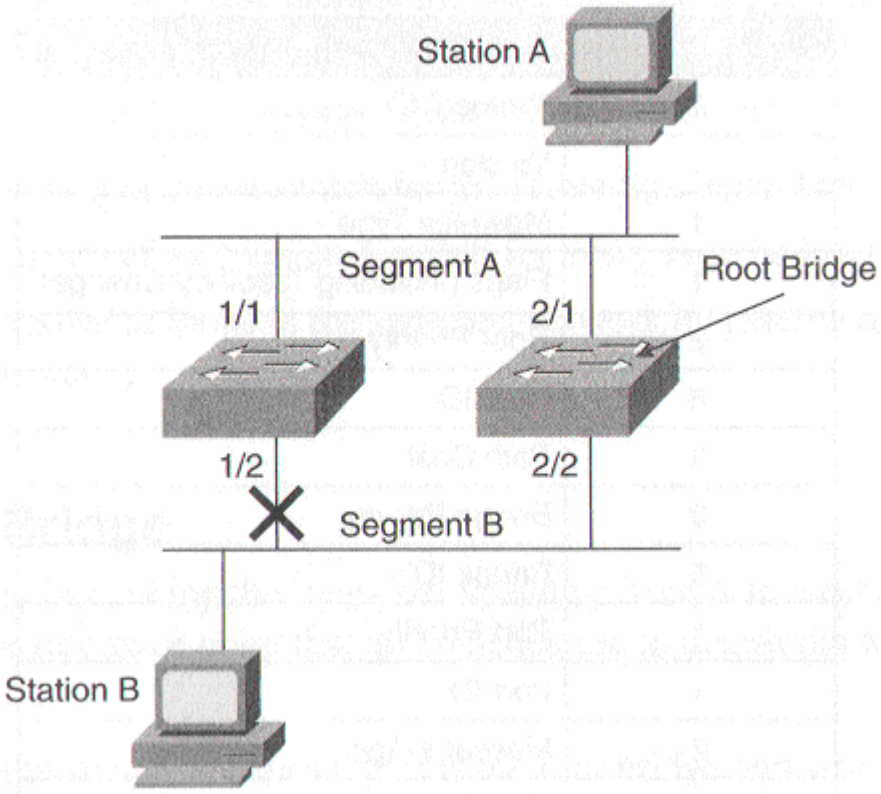
Transparent



Schleife

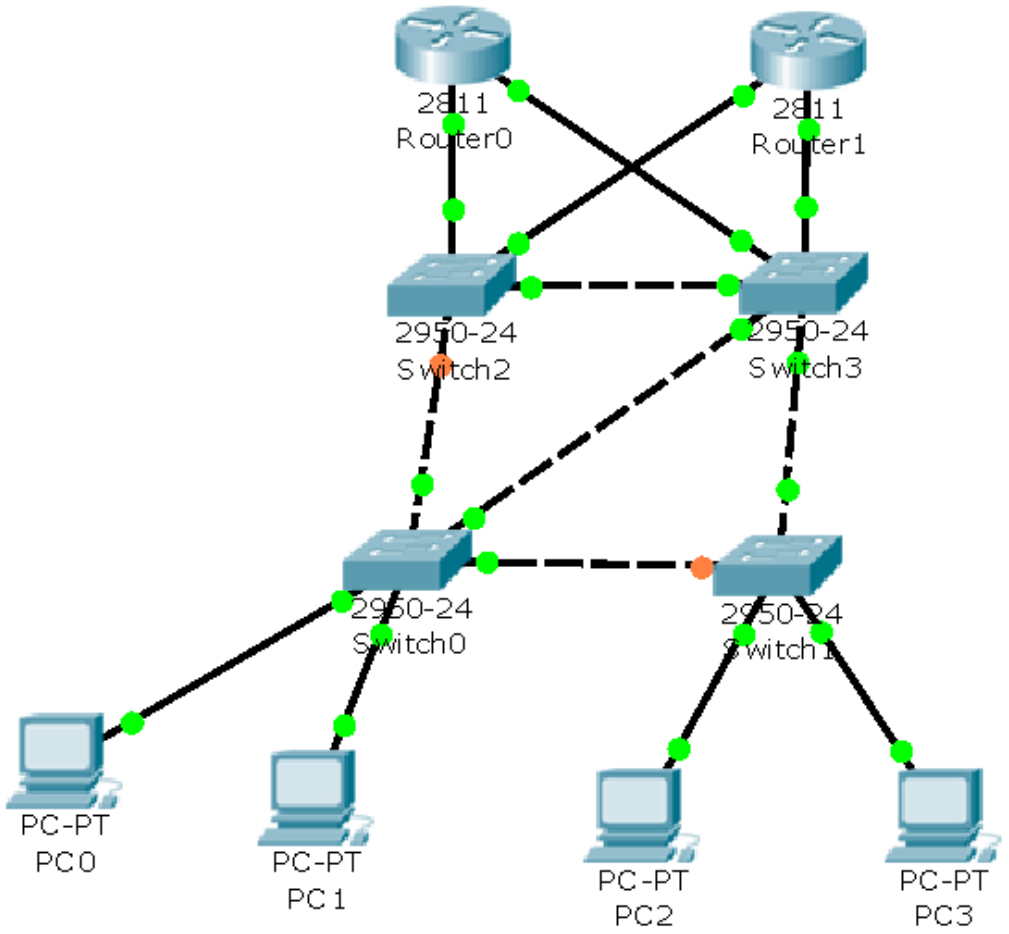
Spanning Tree Protocol

- Dient der **Eliminierung von Schleifen**
- Um das zu erreichen definiert das „Spanning Tree Protokoll“ eine **Baumstruktur, die alle Switches eines Subnetzes umspannt**.
- Es wird eine „**Root Bridge**“ definiert von der aus der Baum aufgespannt wird.



Spanning Tree Protocol im Packet Tracer

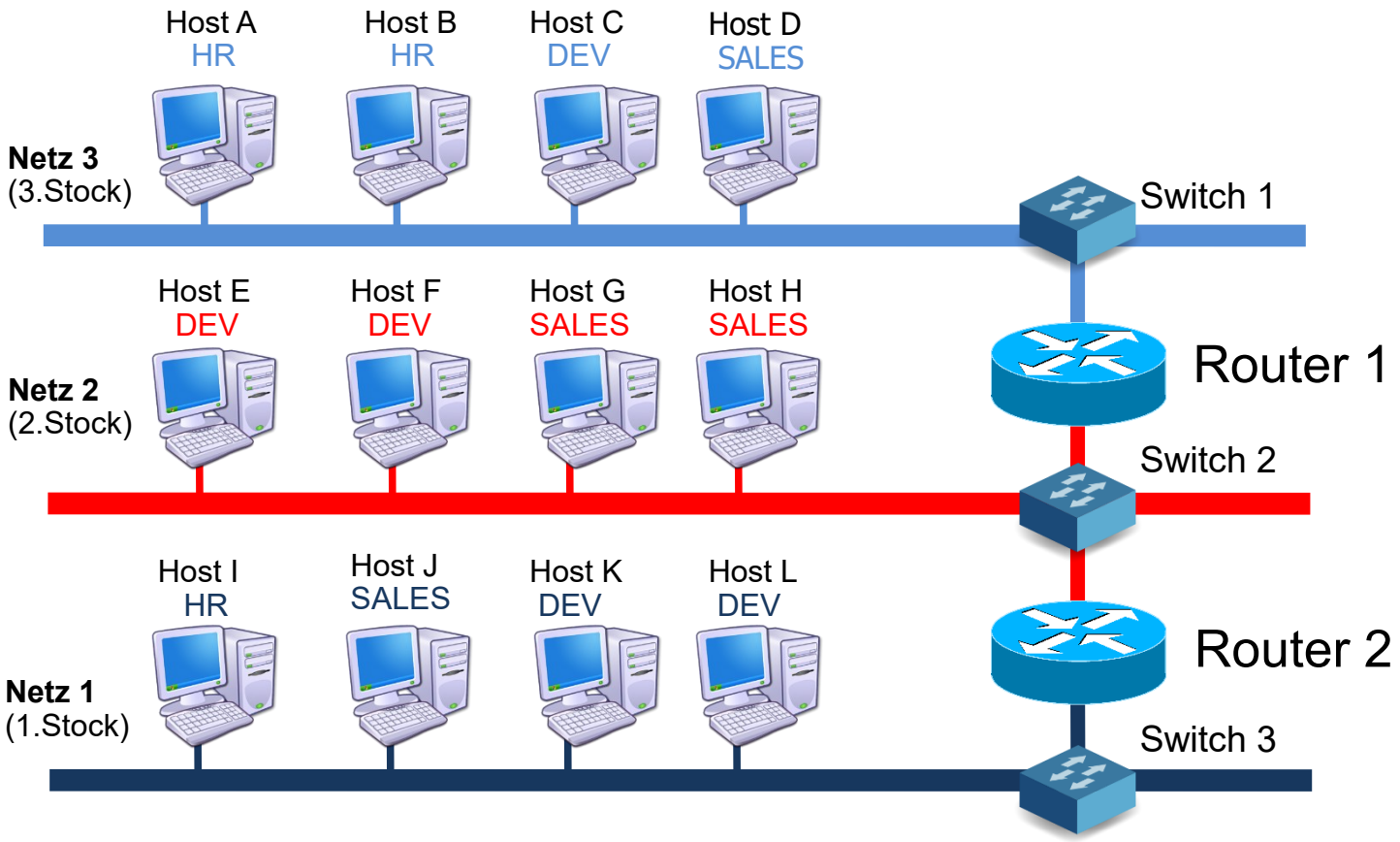
I think that I shall never see
A graph more lovely than a tree.
A tree whose crucial property
Is loop-free connectivity.
A tree that must be sure to span
So packets can reach every LAN.
First, the root must be selected.
By ID, it is elected.
Least cost paths from root are traced.
In the tree, these paths are placed.
A mesh is made by folks like me,
Then bridges find a spanning tree.
— Radia Perlman



Orange Verbindungen sind Sperren um Schleifen zu vermeiden!

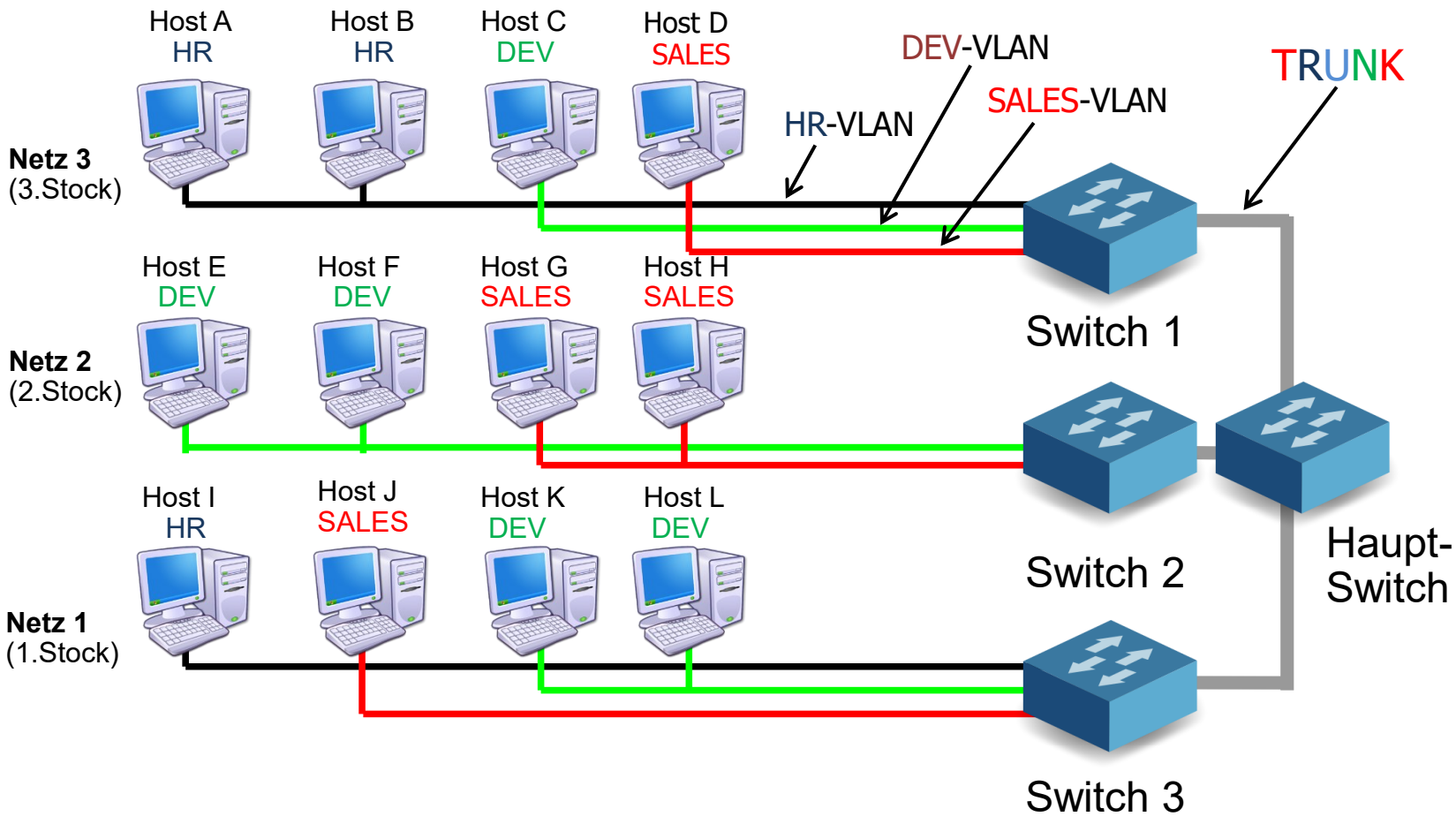
Segmentierung mit **Routern**

- Jedes Segment ist ein **individuelles Subnetz unabhängig** von der **Funktion der beteiligten Hosts**
- Das Subnetz ist **durch die physikalische Anordnung** definiert



Segmentierung mit **Switches**

- Mittels **Virtuellen LANs (VLANs)**
- Jedes **VLAN** ist eine **eigene Broadcast-Domain**
- Vorteile:
  - **Security**
  - **Segmentierung**
  - **Flexibilität**



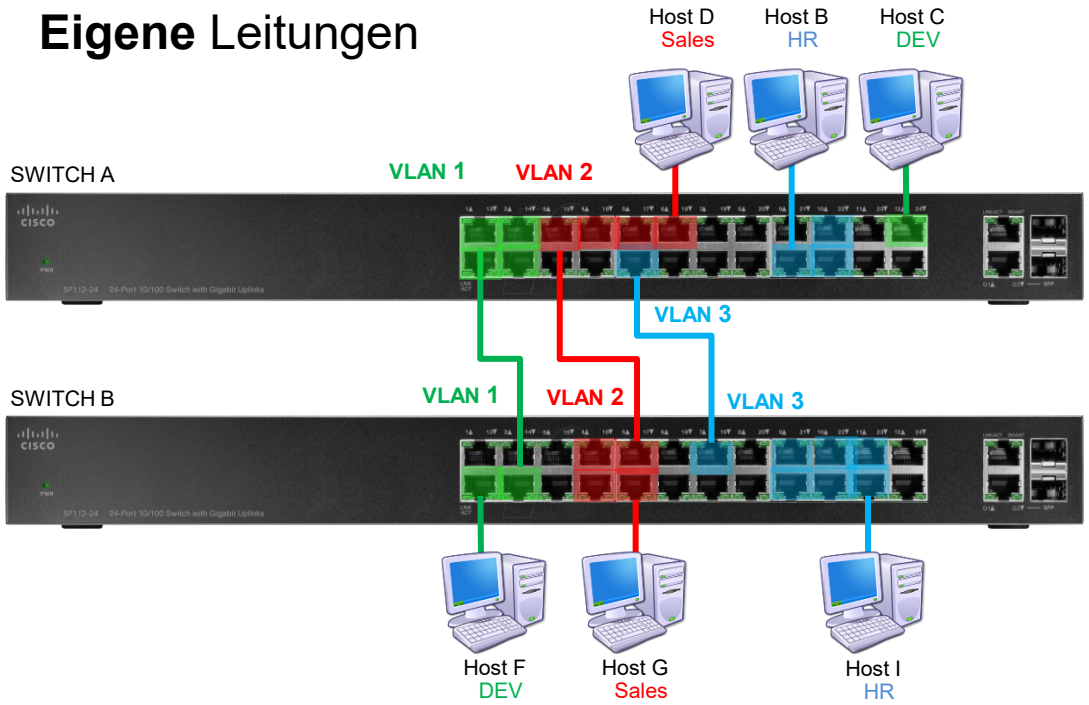
Virtuelle LANs (VLANs)

- Ermöglicht eine **Gruppierung von Benutzern in Broadcast-Domains** unabhängig von der physikalischen Anordnung
- Die **Zuordnung** zu einem VLAN erfolgt rein **durch Konfiguration am SWITCH**
- **Logische Zuordnung** zu Gruppen (Funktion, Applikation, Projektteams)

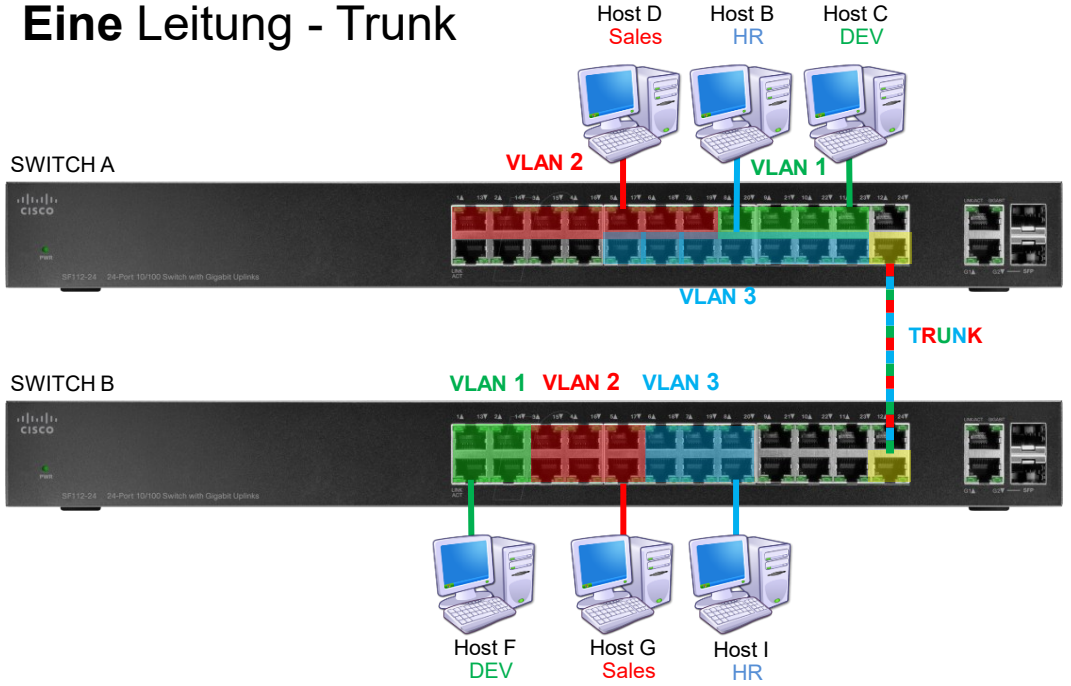
- Jedes **SWITCH-Port** kann **nur einem VLAN** zugeordnet werden.
- Ein VLAN kann sich **über mehrere SWITCHes ausdehnen**
- VLAN-Zuordnung erfolgt auf **Layer 2** (Layer 3 unabhängig)
- **VLAN-VLAN Kommunikation** erfordert ein **Layer-3 Protokoll und Routing**!

Varianten

Eigene Leitungen



Eine Leitung - Trunk



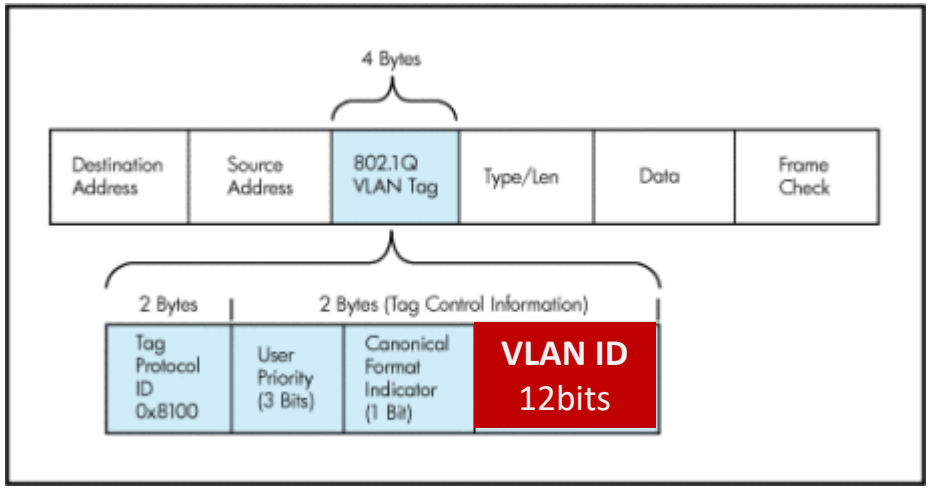
DEV (Development) ... VLAN1: 192.168.1.0/24

Sales (Verkauf) ... VLAN2: 192.168.2.0/24

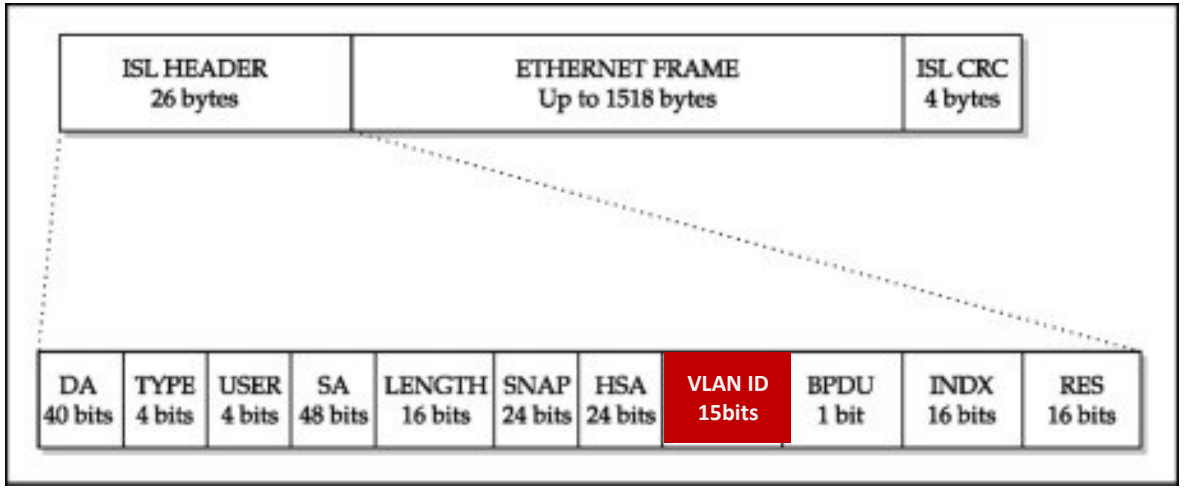
HR (Human Resources) ... VLAN3: 192.168.3.0/24

VLAN Protokolle: **IEEE 802.1q** & **Inter Switch Link**

"dot1q" (IEEE Standard, erweitert Frame)

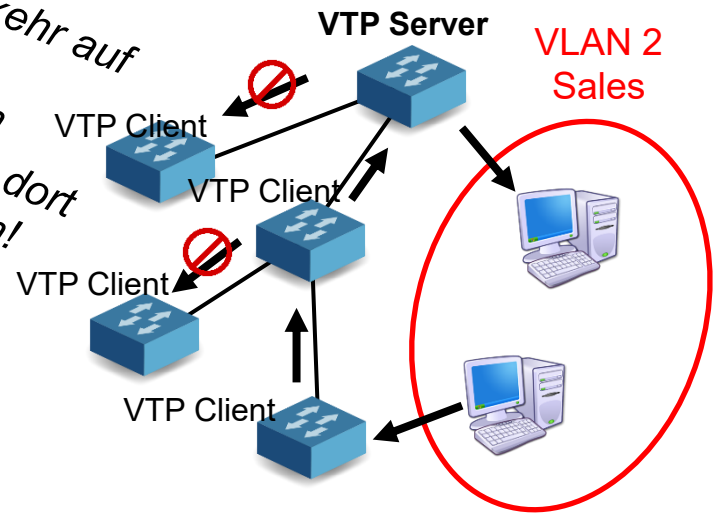


"ISL" (Cisco proprietär, Frame nicht modifiziert)



VLAN Protokolle: **DTP** & **VTP**

VTP Pruning* Minimiert den Verkehr auf Trunk Lines, indem Broadcasts nur dann darüber gehen, wenn dort VLANs dafür existieren!

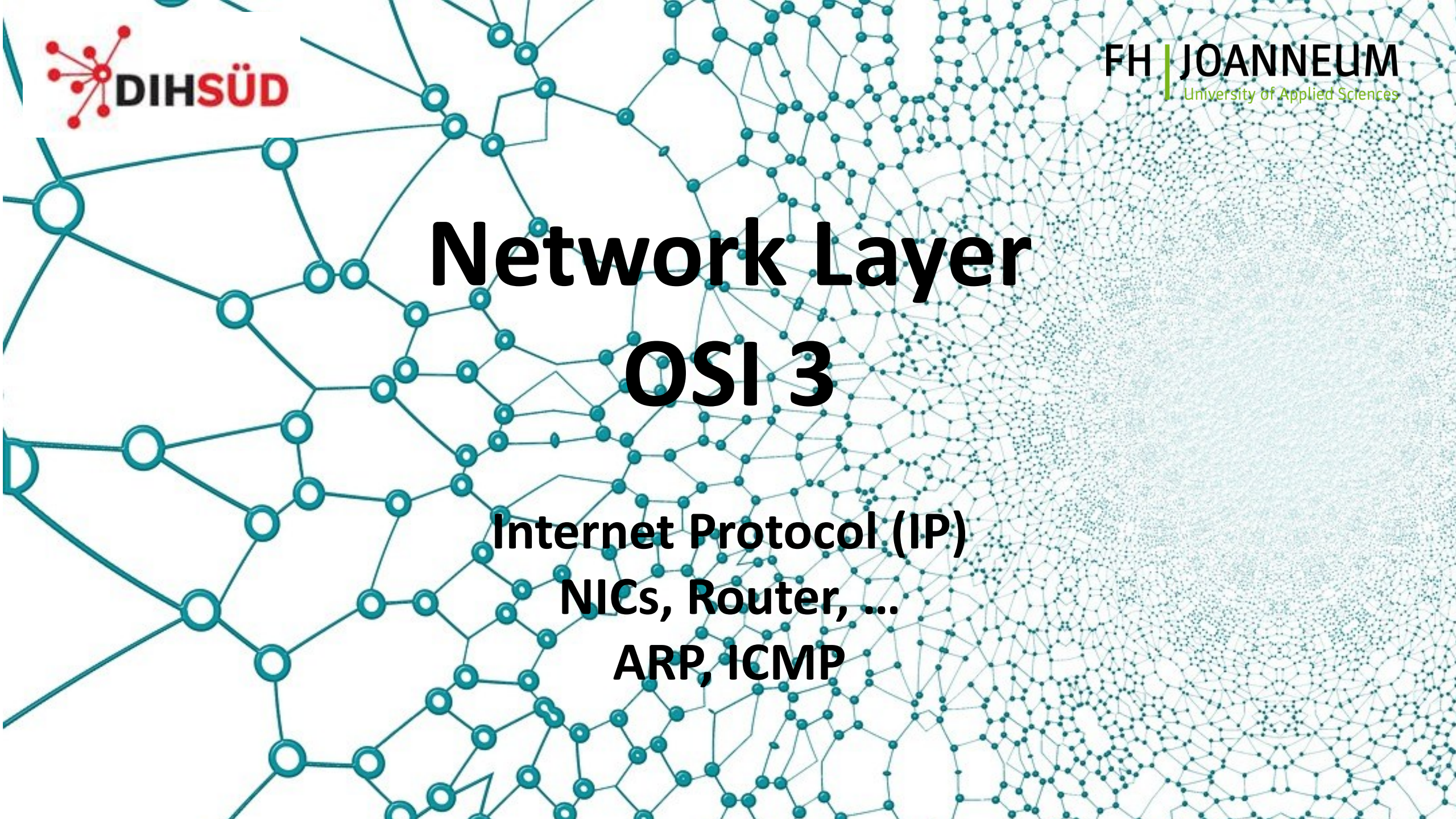


- **Dynamic Trunking Protocol** (DTP)

  - Standardmäßig in Cisco Switches aktiviert
  - **Erkennt**, wenn **am anderen Ende ein Trunk** liegt und
  - verhandelt **Modalitäten** des **Trunking** (z.B.: dot1q oder ISL)
  - und konfiguriert den Port dementsprechend um

- **VLAN Trunking Protocol** (VTP)
  - Wenn man **eine Konfiguration (VTP Domain) auf mehreren Switches** ausrollen will
  - dann definiert man einen Switch zum **VTP-Server** und die anderen zu **VTP-Clients**, die dessen Konfiguration übernehmen
  - Ist ein Switch im **"Transparent" Modus** leitet er die VTP Information zwar weiter, übernimmt sie aber nicht!

(Reduktion)*



DIHSÜD

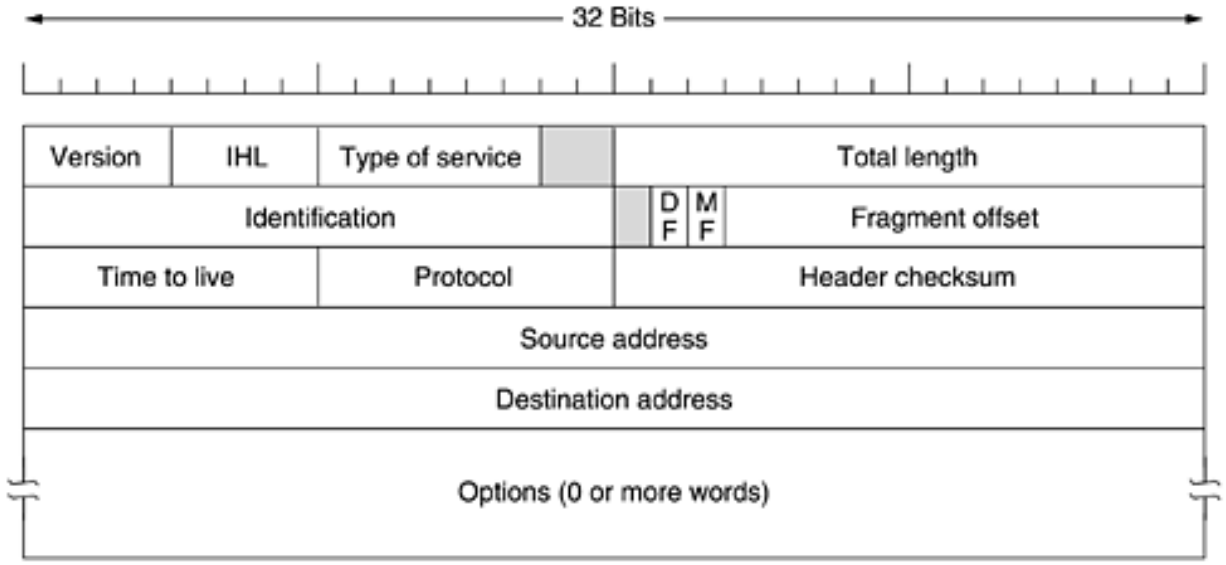
FH JOANNEUM
University of Applied Sciences

Network Layer
OSI 3

Internet Protocol (IP)
NICs, Router, …
ARP, ICMP

IPv4 Protokoll

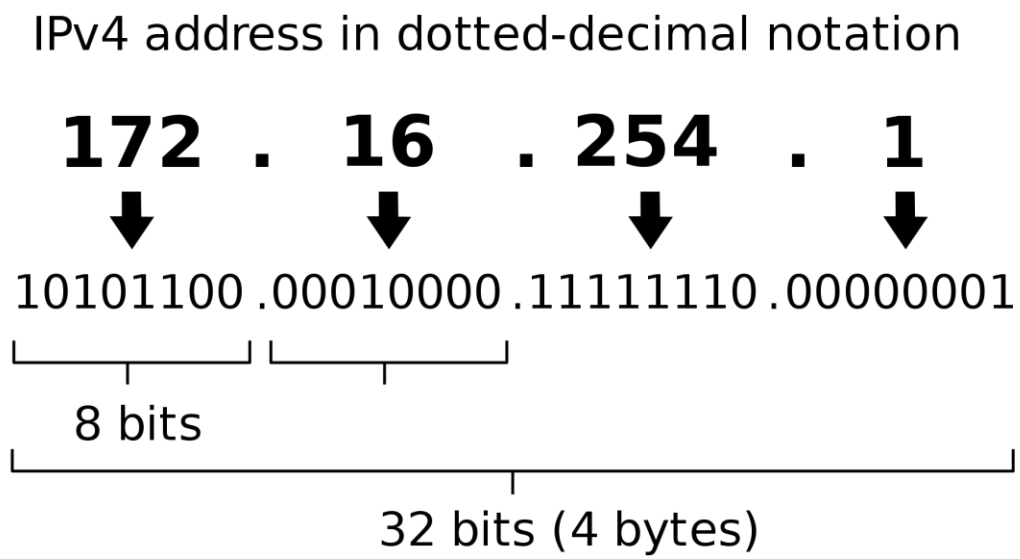
- **Version**
- Header Länge*
- Dienstart
- Größe des Datagramms
- Identifikation für Fragmente
- Entweder hat das Paket keine oder mehrere Fragmente
- Transport Prozess Protokoll

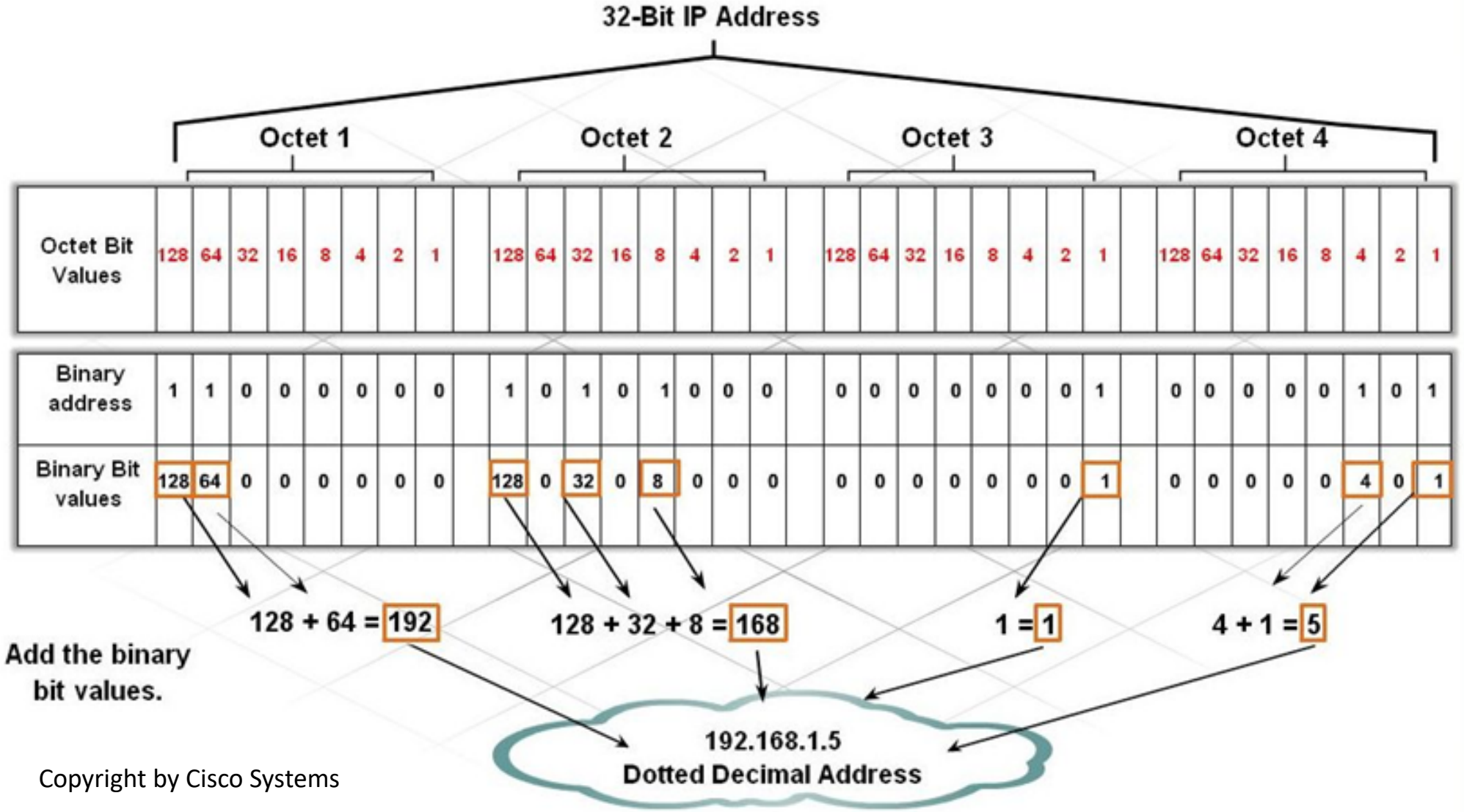



```
Internet Protocol Version 4, Src: 10.52.200.203, Dst: 40.101.76.130
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 40
    Identification: 0x2802 (10242)
  > Flags: 0x4000, Don't fragment
    Time to live: 128
    Protocol: TCP (6)
    Header checksum: 0x8ae7 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.52.200.203
    Destination: 40.101.76.130
```

Adressierung in IPv4

- Die IP Adresse besteht aus einem **32 Bit** Integer Wert
- Aus Gründen der Lesbarkeit werden die **vier Byte Blöcke** der IPv4 Adresse **dezimal dargestellt**.
- Beispiel: 89.105.0.1 =
  - 1500053505 (dez)
  - 59 69 00 01 (hex) oder
  - 10110010110100100000000000000001 (bin)
- Jede reguläre IP Adresse besteht aus einem **Host**anteil und einem **Netzwerk**anteil





Subnetzmasken

192.168.0.1 /24

Netzwerkanteil – **Netzwerk 192.168.0.0** + Hostanteil .1

Berechnung: logisches Und auf die Bits der Adresse mit der gegebenen Subnetzmaske:

```
Network                          | Host
---------------------------------+---------
11000000 10101000 00000000 | 00000001  address
11111111 11111111 11111111 | 00000000  netmask   /24 or
---------------------------------+---------        255.255.255.0
11000000 10101000 00000000 | 00000000  result
```

```
Address: 192.168.0.1 11000000.10101000.00000000 .00000001
Netmask: 255.255.255.0 = 24 11111111.11111111.11111111 .00000000
Wildcard: 0.0.0.255 00000000.00000000.00000000 .11111111
=> Network: 192.168.0.0/24 11000000.10101000.00000000 .00000000 (Class C)
Broadcast: 192.168.0.255 11000000.10101000.00000000 .11111111
HostMin: 192.168.0.1 11000000.10101000.00000000 .00000001
HostMax: 192.168.0.254 11000000.10101000.00000000 .11111110
Hosts/Net: 254 (Private Internet)
```

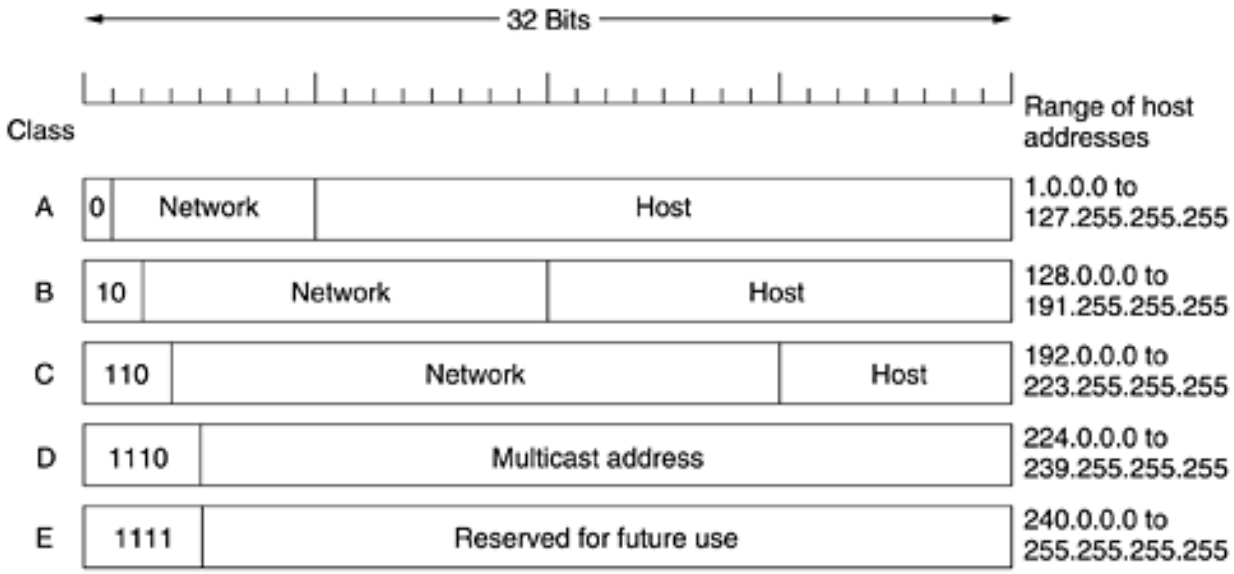
Berechnung: http://jodies.de/ipcalc

Subnet mask quick reference

Host Bit length	math	Max hosts	Subnet mask	Mask octet	Binary mask	Mask length	Subnet length
0	2^0=	1	255.255.255.255	4	11111111	32	0
1	2^1=	2	255.255.255.254	4	11111110	31	1
2	2^2=	4	255.255.255.252	4	11111100	30	2
3	2^3=	8	255.255.255.248	4	11111000	29	3
4	2^4=	16	255.255.255.240	4	11110000	28	4
5	2^5=	32	255.255.255.224	4	11100000	27	5
6	2^6=	64	255.255.255.192	4	11000000	26	6
7	2^7=	128	255.255.255.128	4	10000000	25	7
8	2^8=	256	255.255.255.0	3	11111111	24	8
9	2^9=	512	255.255.254.0	3	11111110	23	9
10	2^10=	1024	255.255.252.0	3	11111100	22	10
11	2^11=	2048	255.255.248.0	3	11111000	21	11
12	2^12=	4096	255.255.240.0	3	11110000	20	12
13	2^13=	8192	255.255.224.0	3	11100000	19	13
14	2^14=	16384	255.255.192.0	3	11000000	18	14
15	2^15=	32768	255.255.128.0	3	10000000	17	15
16	2^16=	65536	255.255.0.0	2	11111111	16	16
17	2^17=	131072	255.254.0.0	2	11111110	15	17
18	2^18=	262144	255.252.0.0	2	11111100	14	18
19	2^19=	524288	255.248.0.0	2	11111000	13	19
20	2^20=	1048576	255.240.0.0	2	11110000	12	20
21	2^21=	2097152	255.224.0.0	2	11100000	11	21
22	2^22=	4194304	255.192.0.0	2	11000000	10	22
23	2^23=	8388608	255.128.0.0	2	10000000	9	23
24	2^24=	16777216	255.0.0.0	1	11111111	8	24

Adressklassen

- Alte Form der Klassifizierung
- Problem: Ungenutzte Adressen in zu großen Bereichen
- Abgelöst durch Classless Inter Domain Routing: Keine strikten Subnetzmasken mehr



Class	Subnetzmaske	Anzahl Netzwerke
A	255.0.0.0	128 (2^7)
B	255.255.0.0	16 384 (2^14)
C	255.255.255.0	2 097 152 (2^21)
D (multicast)	-	-
E (reserviert)	-	-

Spezielle IP Adressen und -Bereiche

- **0.0.0.0 |** bezieht sich auf das aktuelle Netzwerk – diese Adresse wird auch beim Booten verwendet.
- **255.255.255.255 |** Broadcast Adresse, mit der an alle Netzwerke gesendet wird.
  (Oft deaktiviert- und wenn dann mit einem gültigen Netzwerk verwendet!)
- **127.x.x.x |**„Loopback“ Adressierung zum lokalen Testen
- Für die **private Nutzung innerhalb eines LANs**
  (Network Address Translation (NAT) für Verbindung ins Internet benötigt)

CIDR-Notation	Adressbereich	Anzahl Adressen	Historische Netzklassen
10.0.0.0/8	10.0.0.0 – 10.255.255.255	16 777 214	Klasse A: 1 privates Netz mit 16.777.216 Adressen; 10.0.0.0/8
172.16.0.0/12	172.16.0.0 – 172.31.255.255	1 048 574	Klasse B: 16 private Netze mit jeweils 65.536 Adressen; 172.16.0.0/16 bis 172.31.0.0/16
192.168.0.0/16	192.168.0.0 – 192.168.255.255	65 534	Klasse C: 256 private Netze mit jeweils 256 Adressen; 192.168.0.0/24 bis 192.168.255.0/24

Subnetze – Private Adressbereiche

- Für die Nutzung innerhalb eines LANs
  - Network Address Translation (NAT) für Verbindung ins Internet

CIDR-Notation	Adressbereich	Anzahl nutzbarer Adressen
10.0.0.0/8	10.0.0.0 – 10.255.255.255	16 777 214
172.16.0.0/12	172.16.0.0 – 172.31.255.255	1 048 574
192.168.0.0/16	192.168.0.0 – 192.168.255.255	65 534

Address Resolution Protocol (ARP)

- Auflösung von **IP Adresse** (Layer 3) zu **MAC Adresse** (Layer2

Request

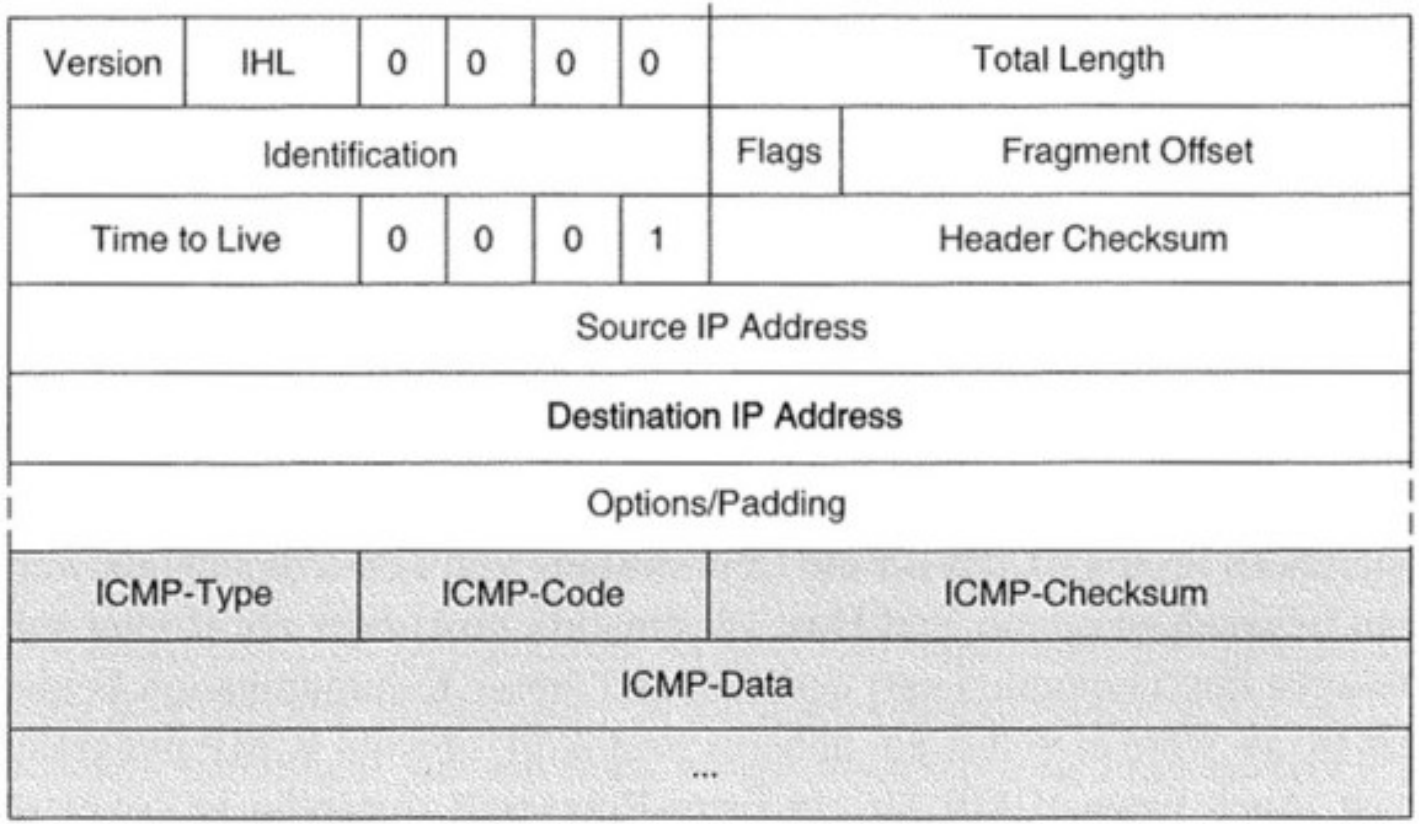
```
Ethernet II, Src: LcfcHefe_60:13:9a (8c:16:45:60:13:9a), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
    Destination: Broadcast (ff:ff:ff:ff:ff:ff)
    Source: LcfcHefe_60:13:9a (8c:16:45:60:13:9a)
    Type: ARP (0x0806)
Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: LcfcHefe_60:13:9a (8c:16:45:60:13:9a)
    Sender IP address: 10.52.200.203
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 10.52.1.254
```

Response

```
Ethernet II, Src: Cisco_ff:fd:2c (00:08:e3:ff:fd:2c), Dst: LcfcHefe_60:13:9a (8c:16:45:60:13:9a)
    Destination: LcfcHefe_60:13:9a (8c:16:45:60:13:9a)
    Source: Cisco_ff:fd:2c (00:08:e3:ff:fd:2c)
    Type: ARP (0x0806)
    Padding: 00000000000000000000000000000000000000
Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: Cisco_ff:fd:2c (00:08:e3:ff:fd:2c)
    Sender IP address: 10.52.1.254
    Target MAC address: LcfcHefe_60:13:9a (8c:16:45:60:13:9a)
    Target IP address: 10.52.200.203
```

Internet Control Message Protocol (ICMP)

- Gekapselt innerhalb des IP
- Ziele:
  - Fehler melden
  - Testen
  - Informationsbeschaffung
- Charakteristika:
  - Bestandteil vom IP
  - Meldet Fehler im Netzwerk, trägt aber nicht zur Verlässlichkeit des Netzwerks bei
  - Eigene Fehler werden nicht berichtet (Damit keine Schleifen entstehen!)

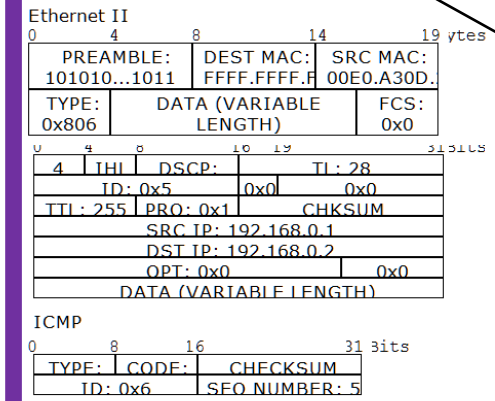


ICMP Nachrichtenfelder

- Nachrichtenfelder
  - Type (8 bit) Basiskategorie / Typ einer Nachricht
  - Code (8 bit) Unterkategorie
  - Checksum (16 bit) Prüfsumme aus Inhaltsfeld
  - Data (Variable Länge) Daten, die zum Beispiel Adressen sein können und zum Packet passen, welches die ICMP Nachricht enthält.
- Typen
  - Ziel nicht erreichbar (3, „Destination unreachable“) – Host oder Port
  - Datenüberlauf der Quelle (4, „Source Quench“) – Puffer sind voll
  - Umleitung (5, „Redirect“) - Info wegen Weiterleitung zu Routern
  - Timeout (11,“Time (to live) exeeded“) – Zeit abgelaufen
  - Konfigurationsproblem (12, „Parameter problem“) – ICMP fehlerhaft

Ping

(ICMP Echo Request)



von  an ???

von **Susi** in **Netzwerk 1**

an **Bastian** in **Netzwerk 1**

Nachricht **an alle im Netzwerk**: Welches **"Symbol*" hat Bastian**?

Bastian's Tabelle

 Susi

Netzwerk1

Susi's Tabelle

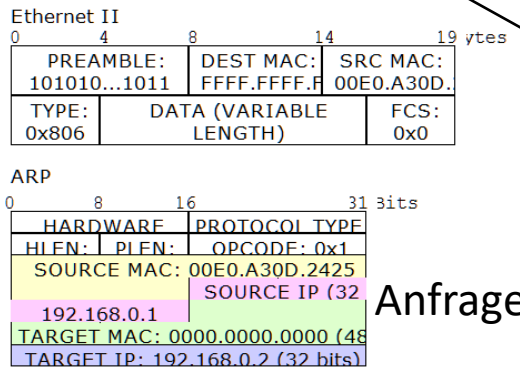
zu Beginn leer

Nachricht **kann nicht gesendet werden**: Das **"Symbol" von Bastian fehlt**!

(Hardwareadresse)*

Symbolanfra

(ARP Request

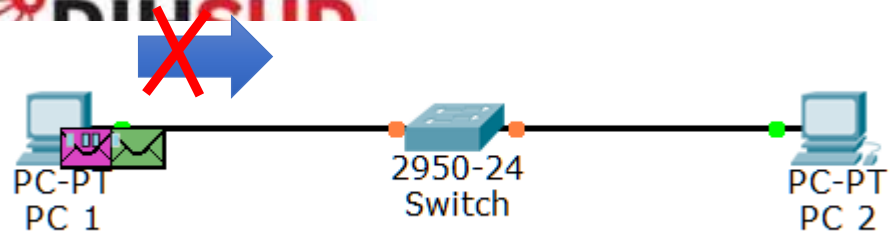


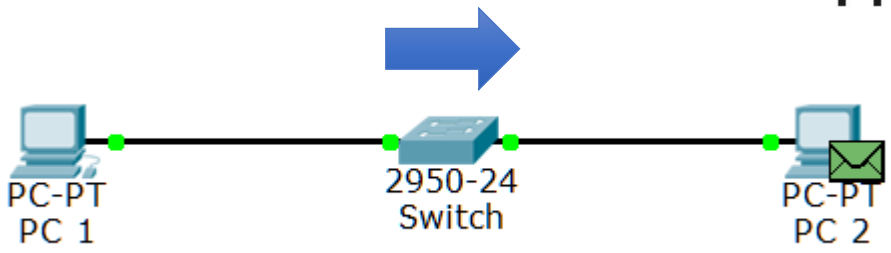
von 

Anfrage

Susi in **N**

Bastian i





Ping (nach IP Adresse 192.168.0.2)

Ethernet II ???

0	4	8	14	19 ytes
PREAMBLE: 101010...1011	DEST MAC:	SRC MAC: 00E0.A30D.		
TYPE: 0x806	DATA (VARIABLE LENGTH)		FCS: 0x0	

IP

0	4	8	16	19	31 3its
4	IHL	DSCP:	TL: 28		
ID: 0x5			0x0	0x0	
TTL: 255	PRO: 0x1	CHKSUM			
SRC IP: 192.168.0.1					
DST IP: 192.168.0.2					
OPT: 0x0				0x0	
DATA (VARIABLE LENGTH)					

von **Susi** in **Netzwerk 1**
an **Bastian** in **Netzwerk 1**

ICMP

0	8	16	31 3its
TYPE:	CODE:	CHECKSUM	
ID: 0x6		SEQ NUMBER: 5	

ARP Request (nach MAC Adresse FF:FF:FF:FF:FF:FF*)

Ethernet II Alle

0	4	8	14	19 ytes
PREAMBLE: 101010...1011	DEST MAC: FFFF.FFFF.F	SRC MAC: 00E0.A30D.		
TYPE: 0x806	DATA (VARIABLE LENGTH)		FCS: 0x0	

ARP

0	8	16	31 3its
HARDWARE		PROTOCOL TYPE	
HLEN:	PLEN:	OPCODE: 0x1	
SOURCE MAC: 00E0.A30D.2425			
	SOURCE IP (32		
192.168.0.1			
TARGET MAC: 0000.0000.0000 (48			
TARGET IP: 192.168.0.2 (32 bits)			

Susi in **Netzwerk 1**

Bastian in **Netzwerk 1**

Schicht mit der Hardware Adressierung

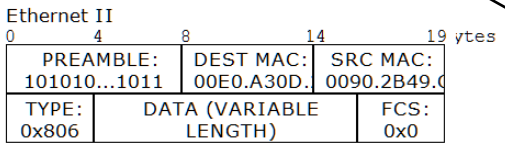
Darüberliegende Schicht mit der logischen Adressierung oder ARP

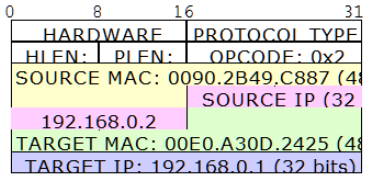
Weil für eine korrekte Adressierung eines "**Ping" Packets** die **Hardwareadresse (MAC Adresse)** fehlt, muss diese vorher erfragt werden- ein **ARP Request** geht an alle im Netzwerk: **"Welche MAC Adresse hat IP 192.168.0.2?"**

*Angenommen **Susi in Netzwerk 1 hat PC1: 192.168.0.1** und **Bastian in Netzwerk 1 hat PC2: 192.168.0.2** (*Broadcast – Nachricht an alle!)*

Symbolantwort

(ARP Response)





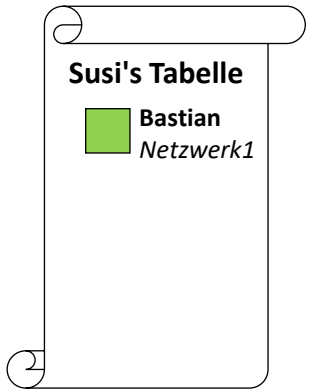
von  an 

Antwort **Bastian** in **Netzwerk 1** 

Susi in **Netzwerk 1** 

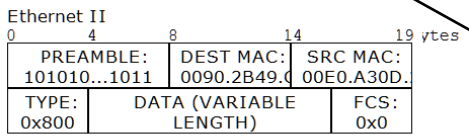
Nun kann die **ursprüngliche Nachricht erst vervollständigt** werden

Bastian sendet an Susi welches **Symbol** er hat!

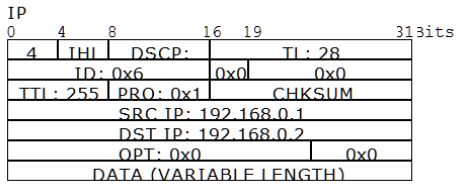


Ping

(ICMP Echo Re

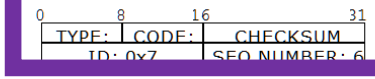


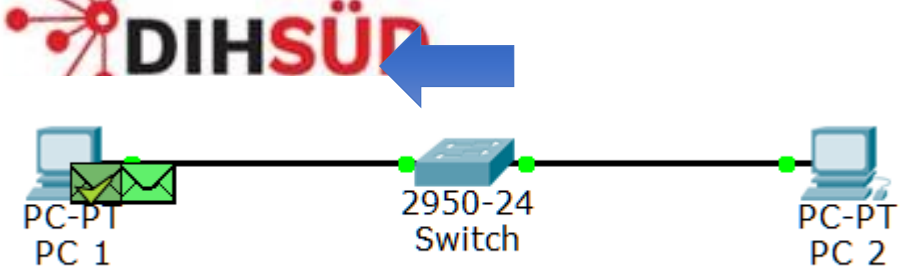
von  ar



von **Susi** in **N**

an **Bastian** i





ARP Response (nach MAC Adresse 00E0.A30D.2425)

Ethernet II

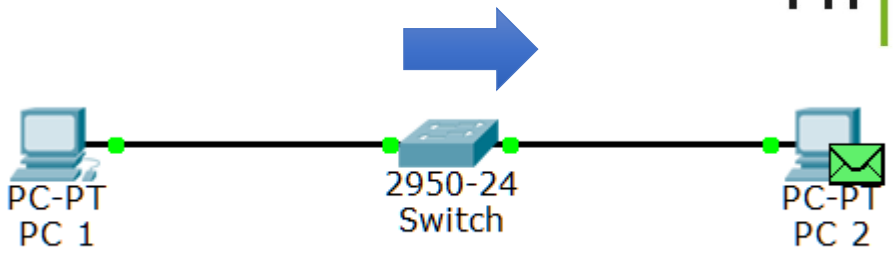
0	4	8	14	19 ytes
PREAMBLE: 101010...1011		DEST MAC: 00E0.A30D.2		SRC MAC: 0090.2B49.C
TYPE: 0x806	DATA (VARIABLE LENGTH)			FCS: 0x0

ARP

0	8	16	31 Bits
HARDWARE		PROTOCOL TYPE	
HLEN:	PLEN:	OPCODE: 0x2	
SOURCE MAC: 0090.2B49.C887 (48			
	SOURCE IP (32		
192.168.0.2			
TARGET MAC: 00E0.A30D.2425 (48			
TARGET IP: 192.168.0.1 (32 bits)			

Bastian in **Netzwerk 1**

Susi in **Netzwerk 1**



Ping (nach IP Adresse 192.168.0.2)

Ethernet II

0	4	8	14	19 ytes
PREAMBLE: 101010...1011		DEST MAC: 0090.2B49.C		SRC MAC: 00E0.A30D.2
TYPE: 0x800	DATA (VARIABLE LENGTH)			FCS: 0x0

IP

0	4	8	16	19	31 Bits
4	IHL	DSCP:	TL: 28		
ID: 0x6			0x0	0x0	
TTL: 255	PRO: 0x1	CHKSUM			
SRC IP: 192.168.0.1					
DST IP: 192.168.0.2					
OPT: 0x0				0x0	
DATA (VARIABLE LENGTH)					

von **Susi** in **Netzwerk 1**
an **Bastian** in **Netzwerk 1**

ICMP

0	8	16	31 Bits
TYPE:	CODE:	CHECKSUM	
ID: 0x7		SEQ NUMBER: 6	

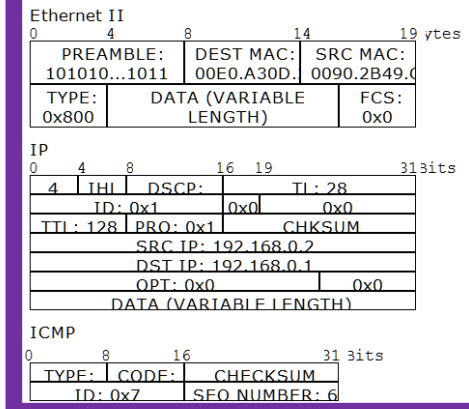
Ping

(ICMP Echo Res

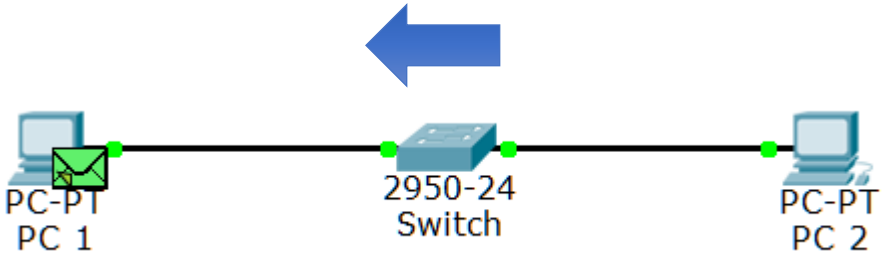
von [ ] ar

von **Bastian**

an **Susi** in **N**



In dieser Nachricht kommt eine **Antwort auf das "Ping"** – die "Echo" Anfrage zurück!



Ethernet II

0	4	8	14	19 ytes
PREAMBLE: 101010...1011		DEST MAC: 00E0.A30D.		SRC MAC: 0090.2B49.(
TYPE: 0x800	DATA (VARIABLE LENGTH)			FCS: 0x0

IP

0	4	8	16	19	31 3its
4	IHL	DSCP:		TL: 28	
ID: 0x1			0x0	0x0	
TTL: 128	PRO: 0x1	CHKSUM			
SRC IP: 192.168.0.2					
DST IP: 192.168.0.1					
OPT: 0x0				0x0	
DATA (VARIABLE LENGTH)					

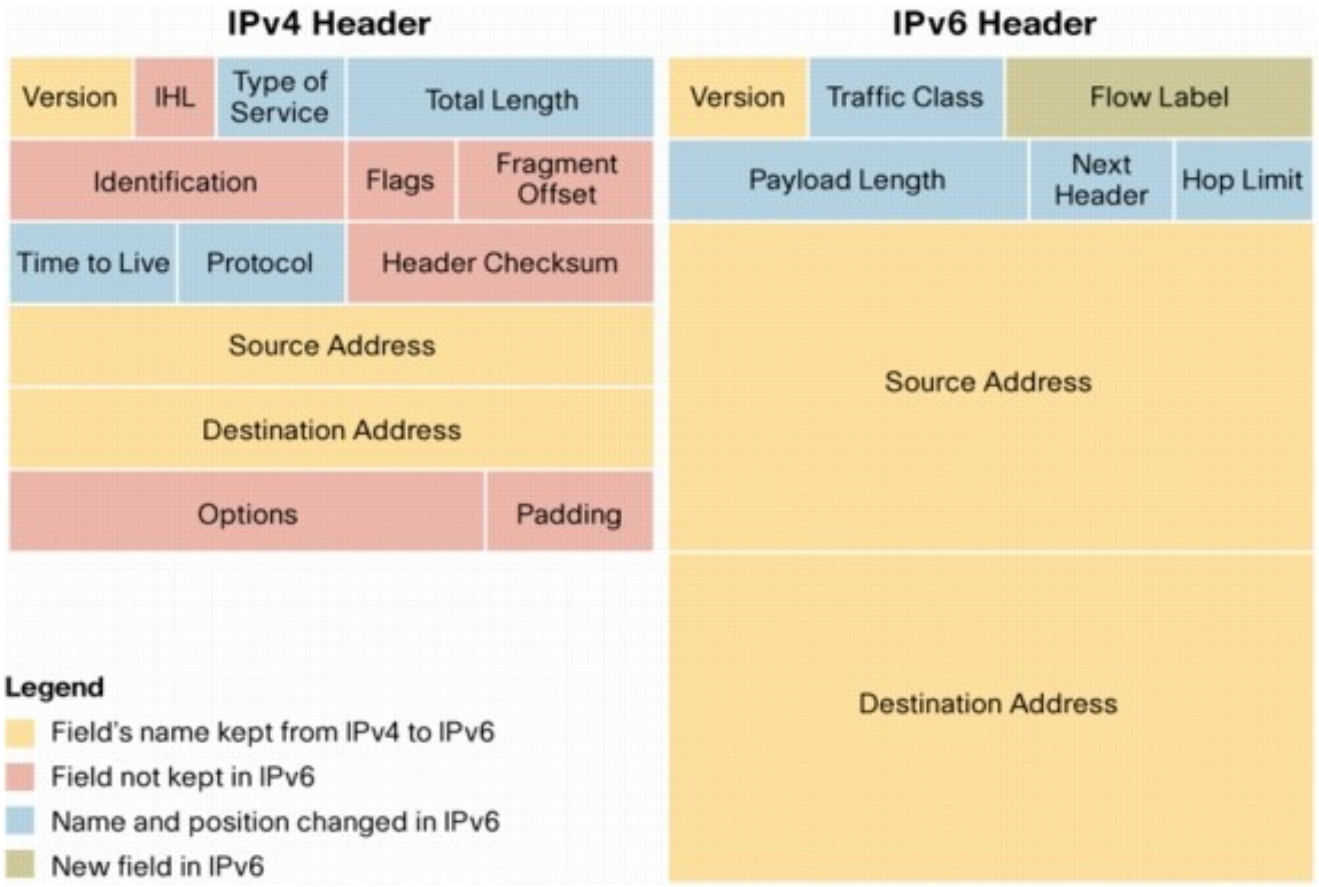
ICMP

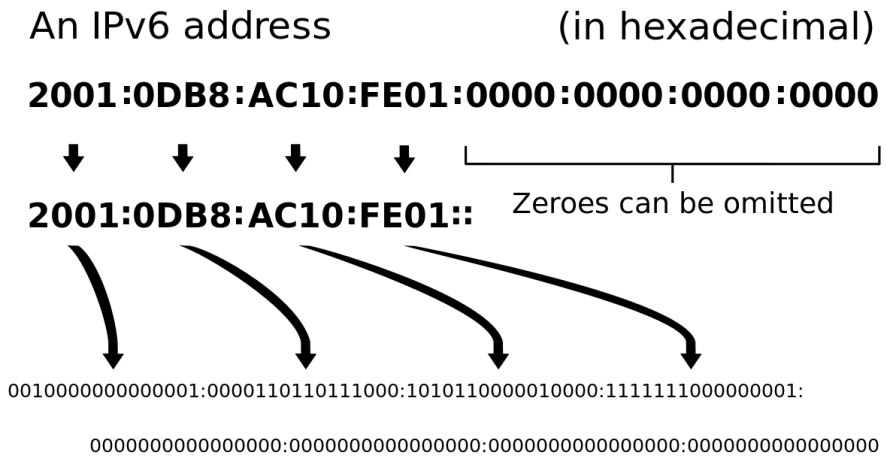
0	8	16	31 3its
TYPE:	CODE:	CHECKSUM	
ID: 0x7		SEQ NUMBER: 6	

ICMP | **Traceroute**

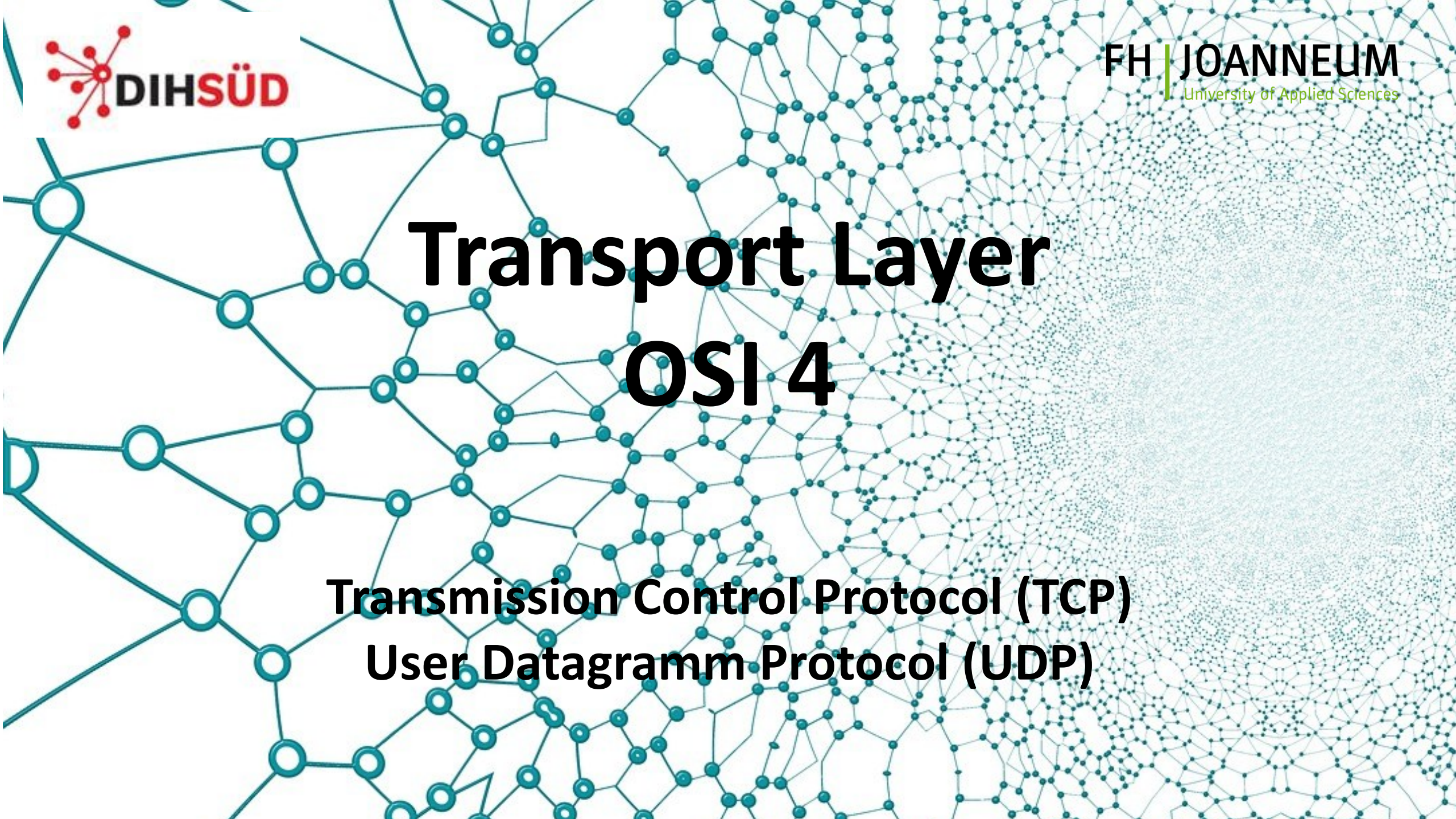
- **Sender versendet ein ICMP Paket mit der TTL = 1**
- **Der nächste Router (Hop) rechnet 1 runter: ICMP „Time exceeded“**
- **Sender versendet ein ICMP Paket mit der TTL = 2**
- **Der übernächste Router (Hop) erreicht TTL = 0: ICMP „Time exceeded“**
- **Dieser Vorgang wird fortgeführt bis:**
  - Das Ziel gefunden wurde oder
  - Keine brauchbare Antwort mehr kommt

IPv4 & IPv6





https://upload.wikimedia.org/wikipedia/commons/thumb/7/70/Ipv6_address_leading_zeros.svg/1200px-Ipv6_address_leading_zeros.svg.png



Transport Layer
OSI 4

Transmission Control Protocol (TCP)
User Datagramm Protocol (UDP)

Transportschicht - Überblick

- „**Ports**“ und andere zusätzliche Informationen werden den Paketen vorangestellt.
- Ports identifizieren **Anwendungen zwischen 2 Kommunikationspartnern**
  (welche durch ihre IP Adressen identifiziert werden)
- Es gibt immer 2 Ports:
  - Der **Quellport** (Source Port)
  - Der **Zielport** (Destination Port)
- Der **Zielport identifiziert üblicherweise den angeforderten Dienst**.

Transmission Control Protocol (TCP)

- Grundidee
  - „Sich nicht um einzelne Netzwerkpakete kümmern sondern das **Empfangen/Senden großer Datenmengen in einer TCP Sitzung** ermöglichen.“
  - Daraus folgt: „Eine Verbindung muss hergestellt und aufrecht erhalten bleiben.“
- TCP verpackt Daten in so genannte **Segmente**.
  - Die Größe der Segmente wird durch darunterliegende Protokolle definiert.
  - Die Anwendungsdaten werden mit einem TCP Header versehen, der alle Informationen enthält, um die Daten sicher und verlässlich zu transferieren.
- **Verbindungsorientiert**
  - **Verbindungs-Management,** Kommunikationsdienst zwischen 2 Partnern und **Datentransport**!
  - **Full-Duplex** (Mehrfachverbindungen möglich)
  - Aber **Kosten beim Datendurchsatz**
  - **Hohen administrative Aufwand** im Protokoll und im Header

Schicht	Dienste / Protokolle / Anwendungen			
Anwendung	HTTP	IMAP	DNS	SNMP
Transport	TCP		UDP	
Internet	IP (IPv4 / IPv6)			
Netzzugang	Ethernet, ...			

Transportschicht im IP Stack

Quell-Port			Ziel-Port
Sequenz-Nummer			
Acknowledgement-Nummer			
D. O.	Res.	Flags	Window-Größe
Check-Summe			Urgent-Pointer
Optionen/Füllbits			
Daten....			

TCP Header

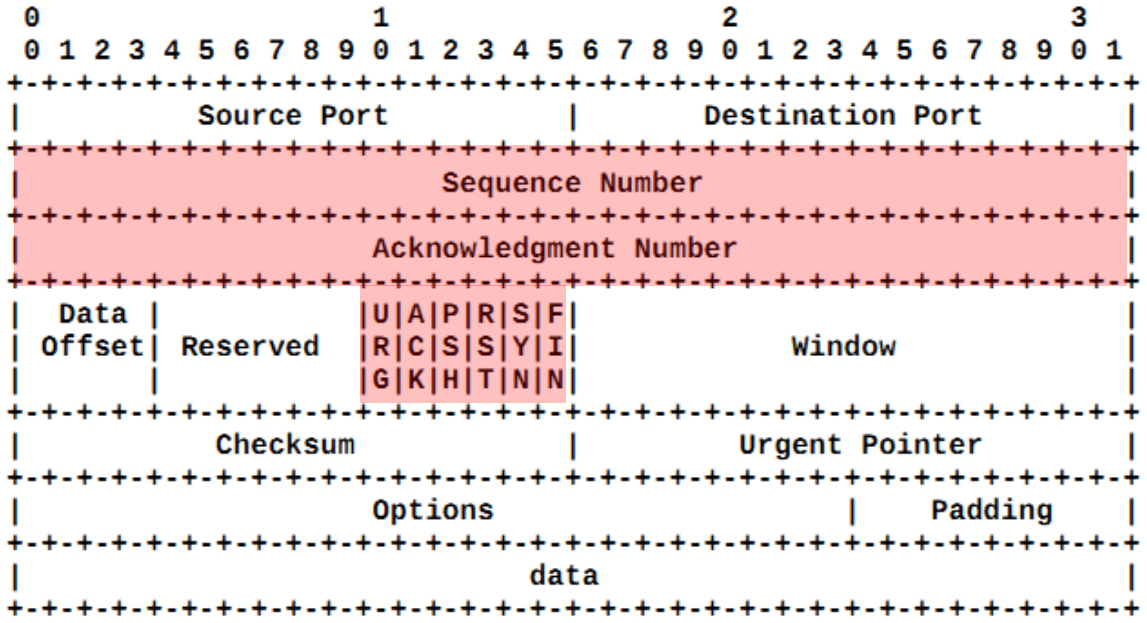
TCP: **Funktionsüberblick**

- **Verbindungs-Management**
  Aufbau, Überwachung und Abbau einer Verbindung
- **Verlässliche Kommunikation**
  Keine Pakete gehen verloren.
- **Fehlererkennung**
  Daten bleiben unverändert. (Integrität!)
- **Flußkontrolle**
  Übertragungsrate wird den Übertragungskanal angepasst.
- **Überlastungsschutz**
  Paketstau in Netzen mit viel Verkehr/Netzlast wird vermieden.
- **Zeitüberwachung**
  Bei Zeitüberschreitung wird eine Verbindung aufgelöst.

TCP: **Verbindungsmanagement**

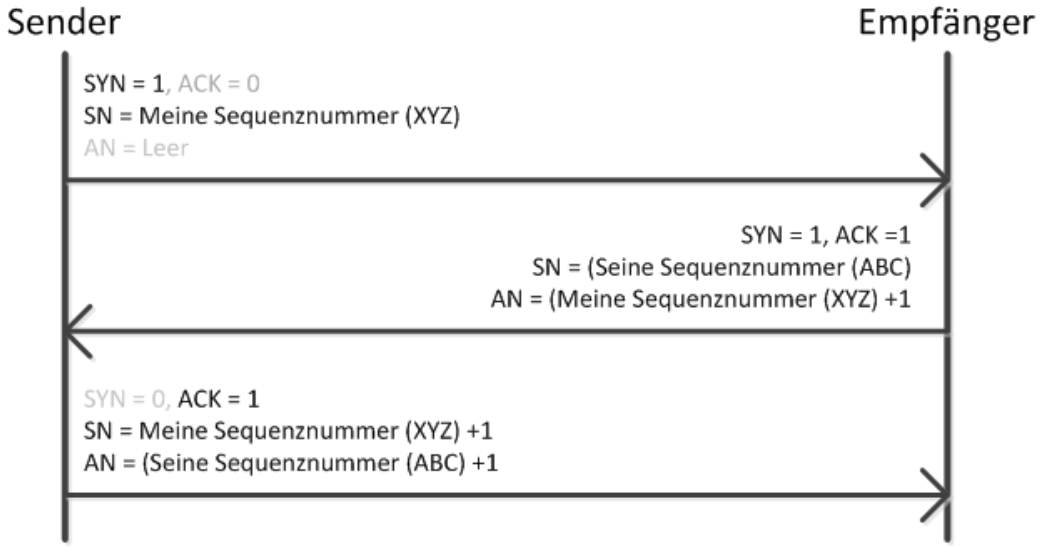
- Sequenznummer & Bestätigungsnummer

  Wichtig ist, dass sich beide Kommunikationspartner auf eine Sequenznummerierung einigen. Daher sind diese Bereiche für die Verbindung wichtig.
- Flags
  - URG (Urgent Pointer): Daten mit Priorität behandeln
  - **ACK** (Acknowledgement): Empfangsbestätigung
  - PSH (Push): Daten müssen sofort behandelt werden.
  - **RST** (Reset): Verbindung wegen Fehler beendet.
  - **SYN** (Synchronize): Zum Aufbau einer Verbindung verwendet.
  - **FIN** (Final): Ende einer Verbindung.



TCP: **3 Way Handshake** (3 Phasen Verbindungsaufbau)

- **Phase1**:
  SYN und ISN_{Sender} im Sequenznummernfeld an Empfänger
- **Phase2**:
  Empfänger sendet SYN, ACK, $ISN_{Empfänger}$ im Sequenznummernfeld und ISN_{Sender} +1 im Bestätigungsnummernfeld zurück.
- **Phase3**:
  ACK und ISN_{Sender} +1 im Sequenznummernfeld und $ISN_{Empfänger}$ +1 im Bestätigungsnummernfeld an Empfänger.



Funktioniert nur, wenn beide Partner für eine Verbindung bereit sind. (Zum Beispiel wartet der Webserver auf eine Anfrage: Passiver, offener Status)

TCP: Verbindungsabbau

- **Ungeplanter Verbindungsabbruch:**
  RST Flag von einem der Partner.
- **Normaler Verbindungsabbau:**
  Gegenseitiges Senden von FIN und Antworten mit ACK Flags.
- **Schneller, normaler Verbindungsabbau:**
  Gleichzeitiges senden von FIN und ACK in einem Paket.

TCP: **Verlässliche Kommunikation**

- Sequenznummern identifizieren Pakete.
- Dazu wird zuerst eine zufällige „Initial Sequence Number“ (ISN) erzeugt.
- Jedes Byte an Daten erhöht die Sequenznummer.
- Der Empfänger muss die Sequenznummern annehmen.
- Dazu sendet der Empfänger die nächst höchste Sequenznummer, die er bekommen hat zurück!

Beispiel:

- ISN = 100
- Das übertragene Segment enthält Bytes mit den Nummern: 100, 101, 102, 103, 104, 105
- Die Nummer 100 wird übertragen.
- Der Empfänger sendet 106 zurück. (= die Nummer des nächsten erwarteten Bytes)

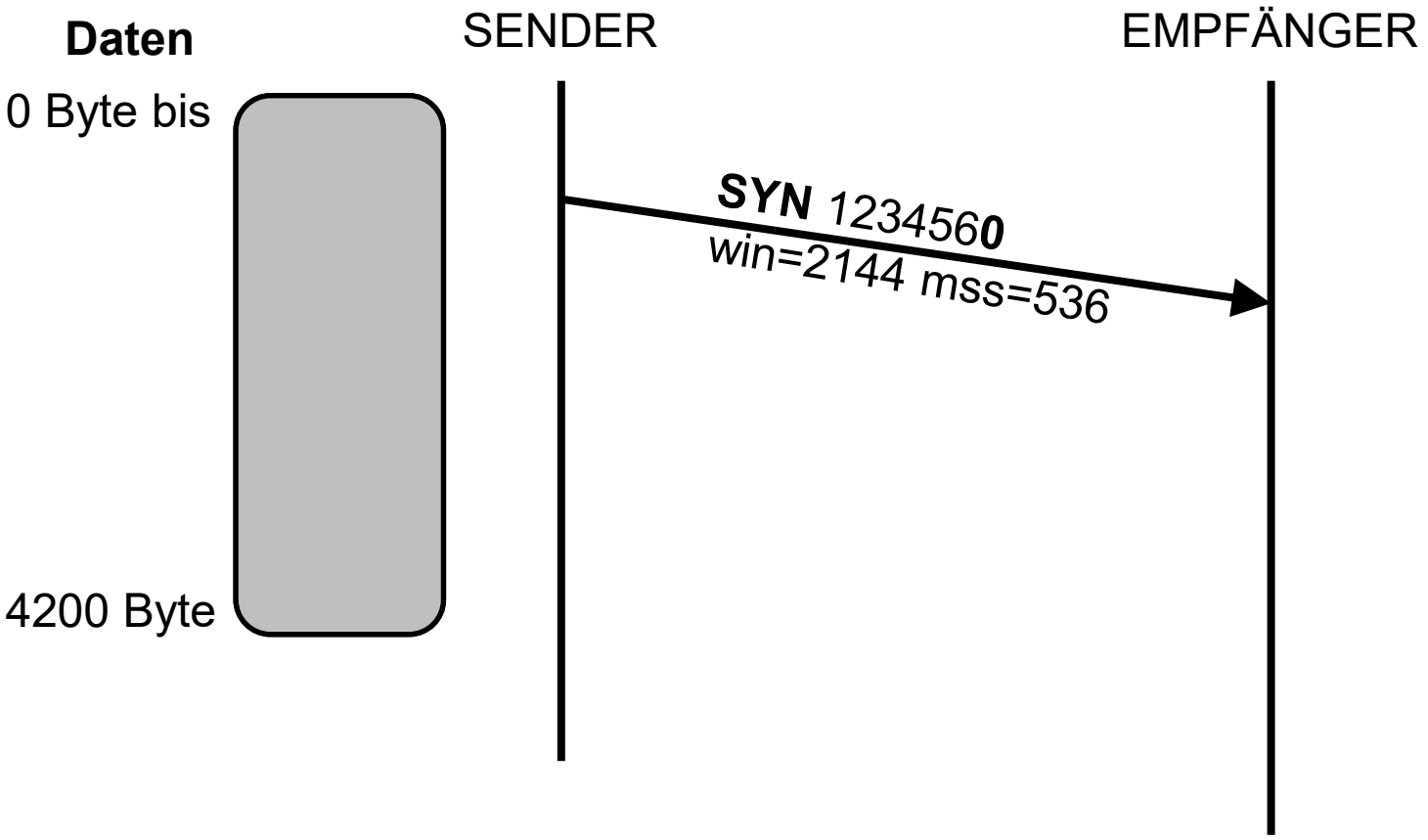
TCP: **Flusskontrolle: Sliding Windows** (Schiebfenster)

- Übertragungsgeschwindigkeit wird gemessen.
- Zeitnahe und verlässliche Bearbeitung der Daten wird ermöglicht.
  Wenn ein Empfänger wenig Daten empfangen kann, wird der Datenfluss herabgesetzt.

„Sliding Windows“ (Schiebefenster) – Funktionsweise:

- Partner bestimmen eine „Fenstergröße“ (***Window Size***) für Daten.
- Der Empfänger kann alle Segmente oder nur das letzte Segment (am besten das des Fensters) bestätigen. (Letzteres= weniger Overhead)
- Fehlende Segmente (oder falls sie nicht zeitgerecht angekommen sind) werden noch mal gesendet.
- Es gibt eine dynamisch bestimmte Zeit (***Round Trip Time – RTT***), in der die Bestätigung da sein muss.

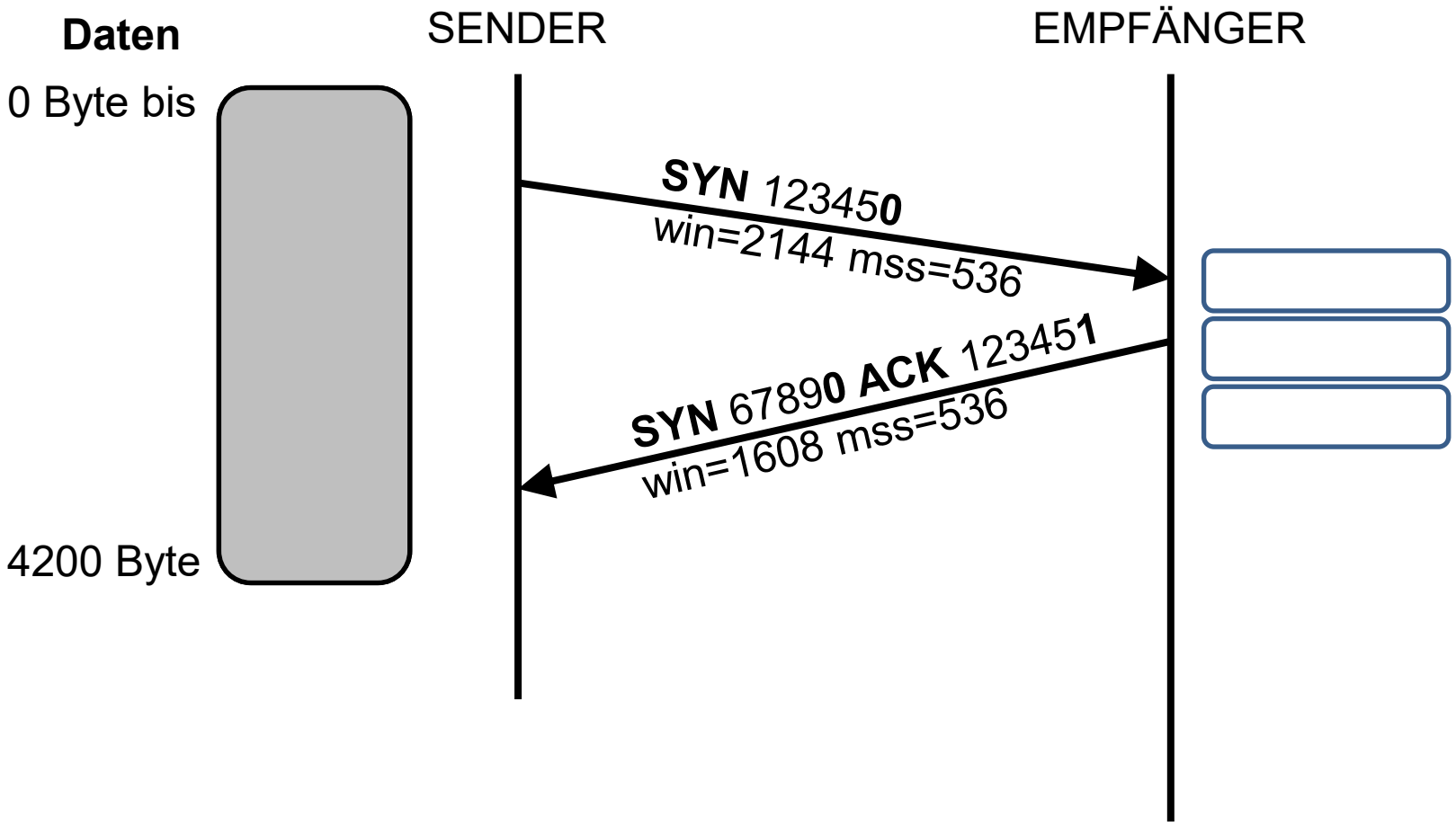
TCP: **Handshake & Sliding Windows** 1/8



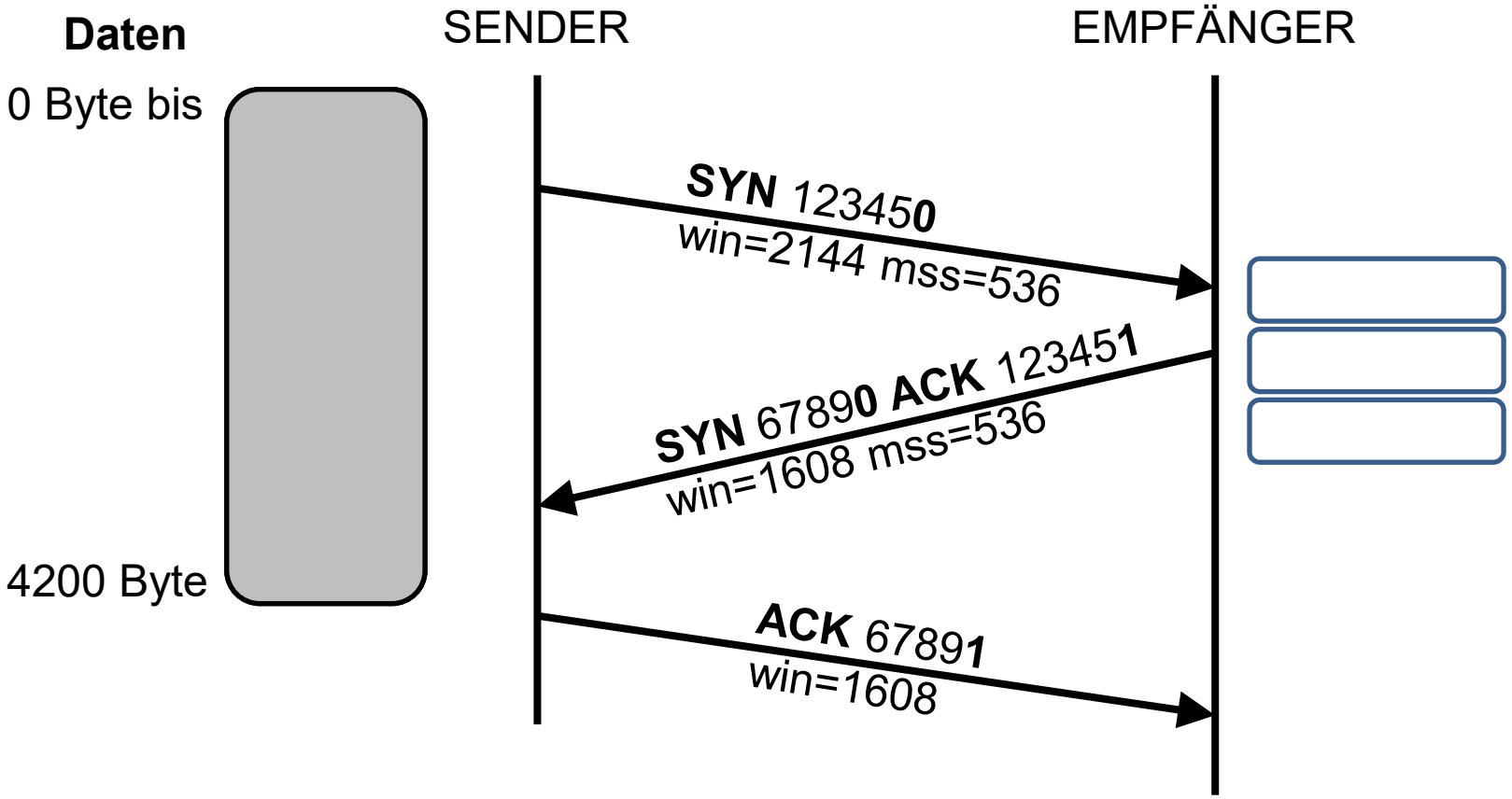
Maximum Segment Size (mss)

Maximale Anzahl von Bytes, die in einem Segment als Nutzdaten versendet werden können.

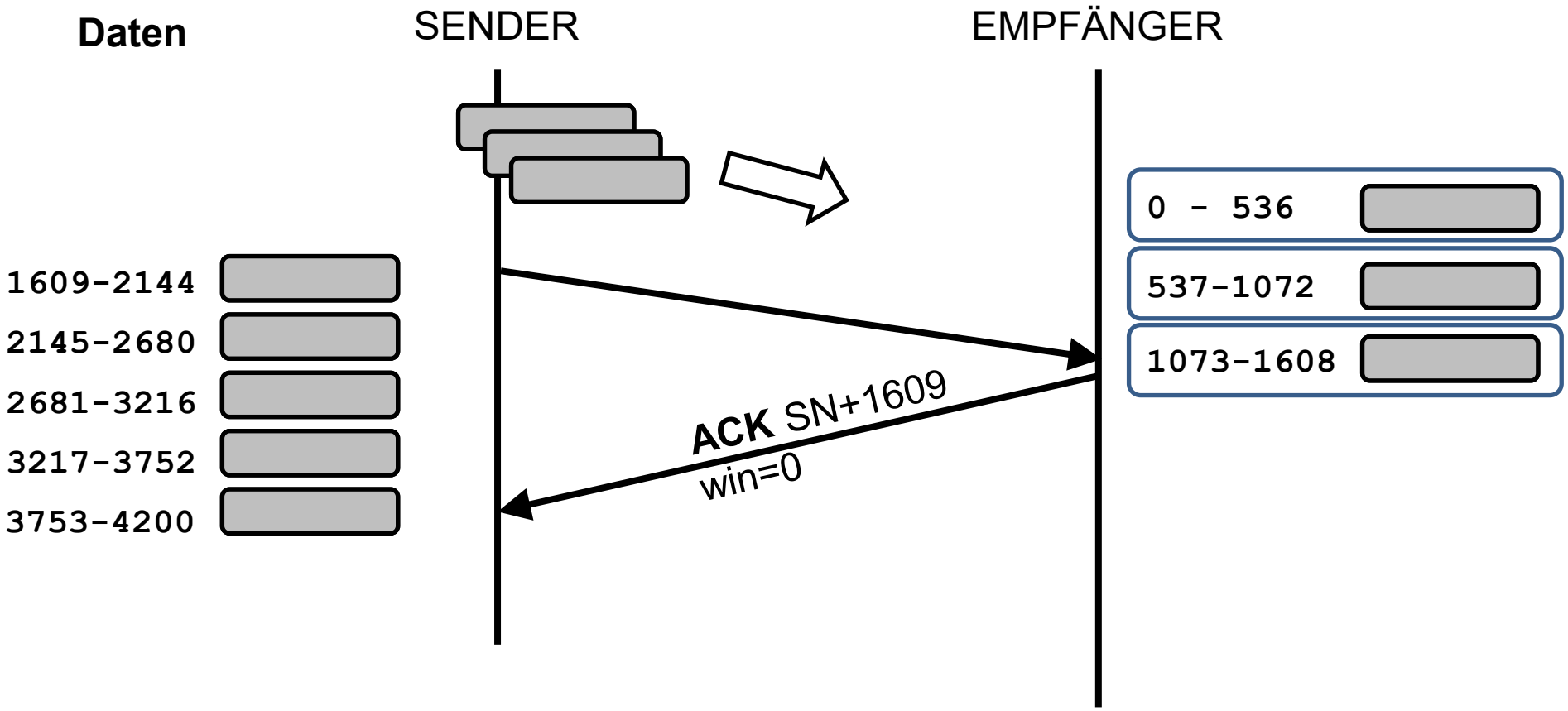
TCP: **Handshake & Sliding Windows** 2/8



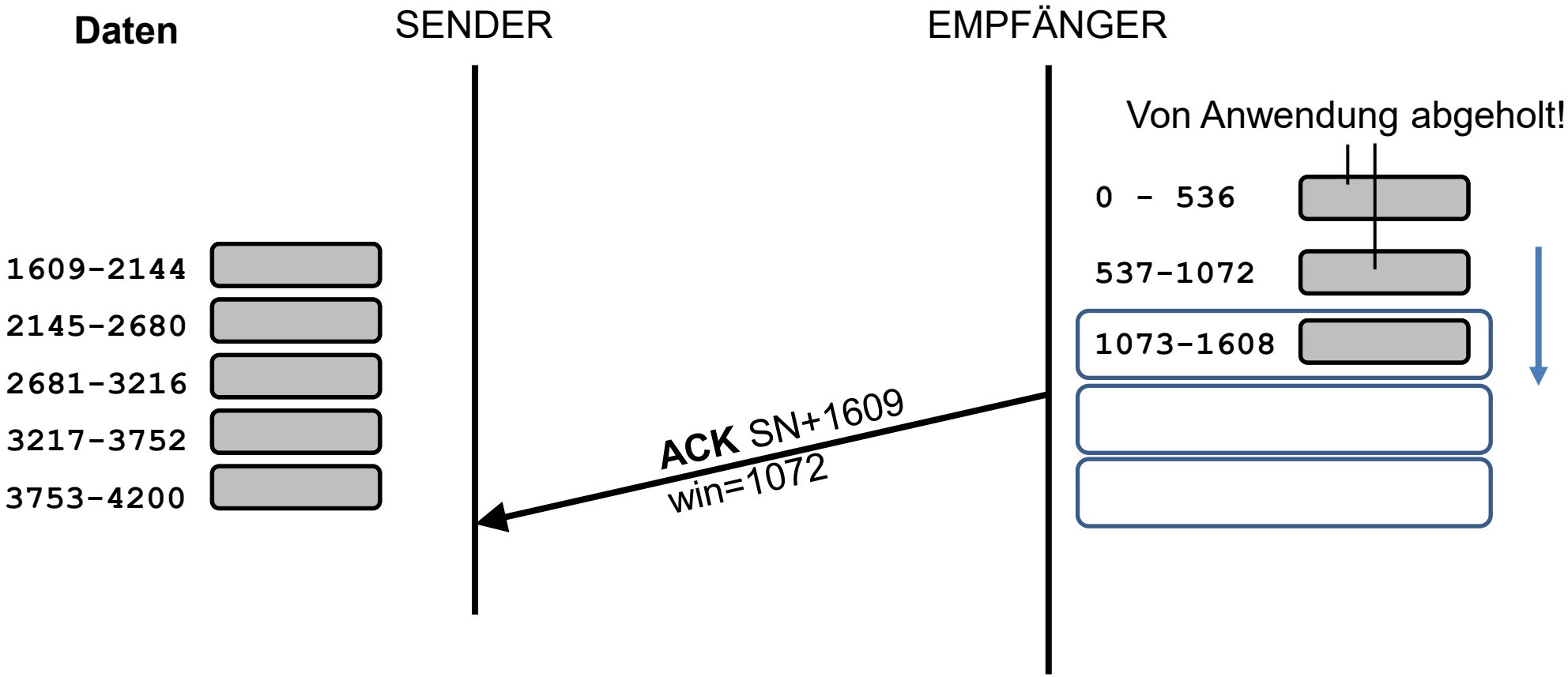
TCP: **Handshake & Sliding Windows** 3/8



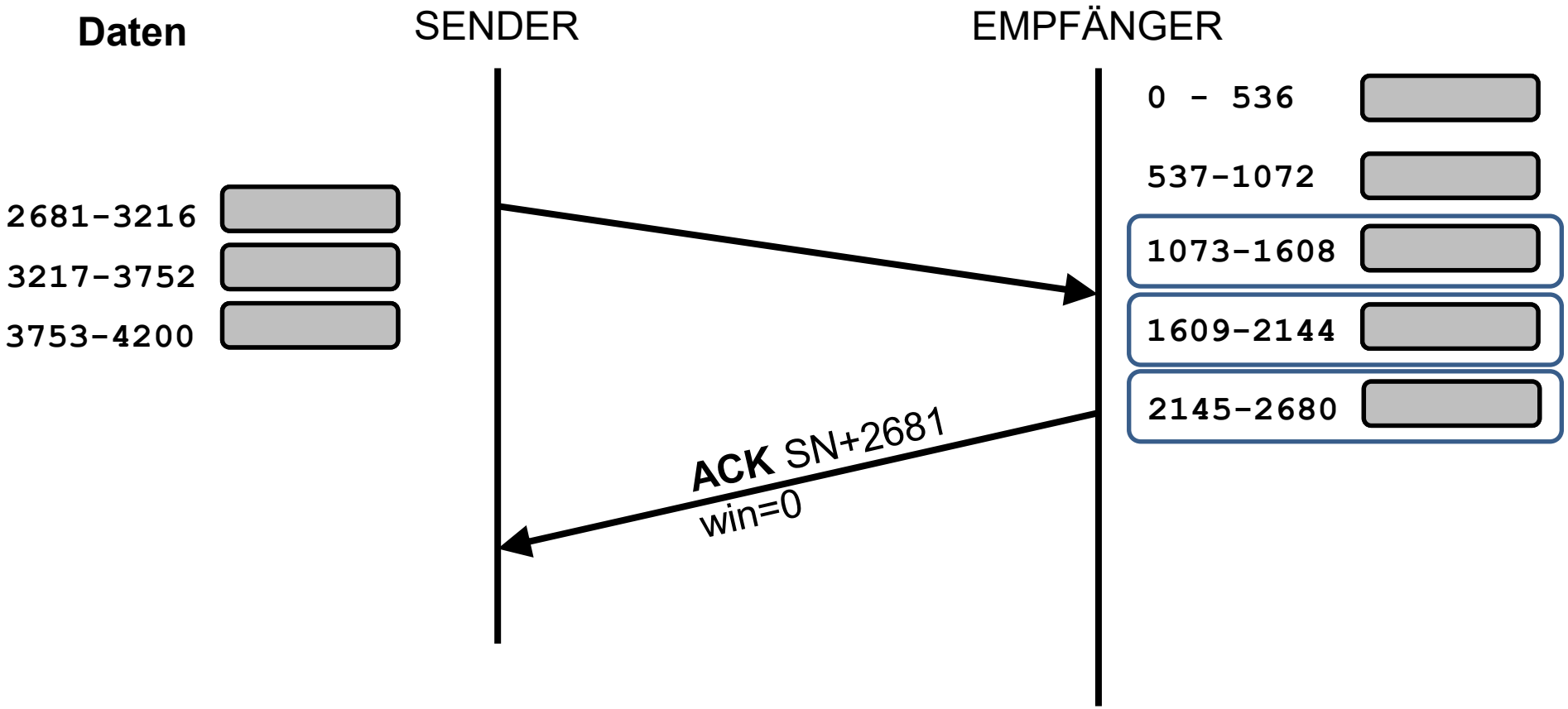
TCP: **Handshake & Sliding Windows** 4/8



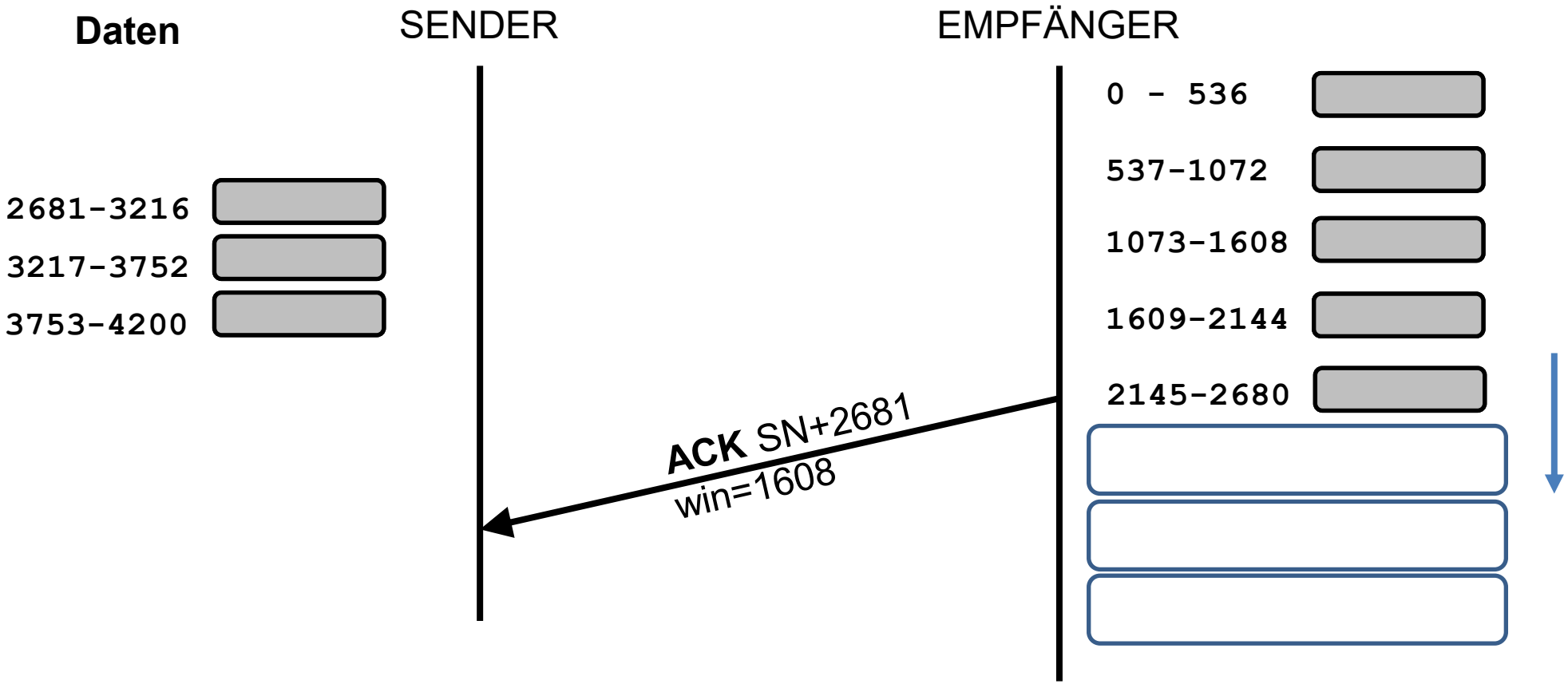
TCP: **Handshake & Sliding Windows** 5/8



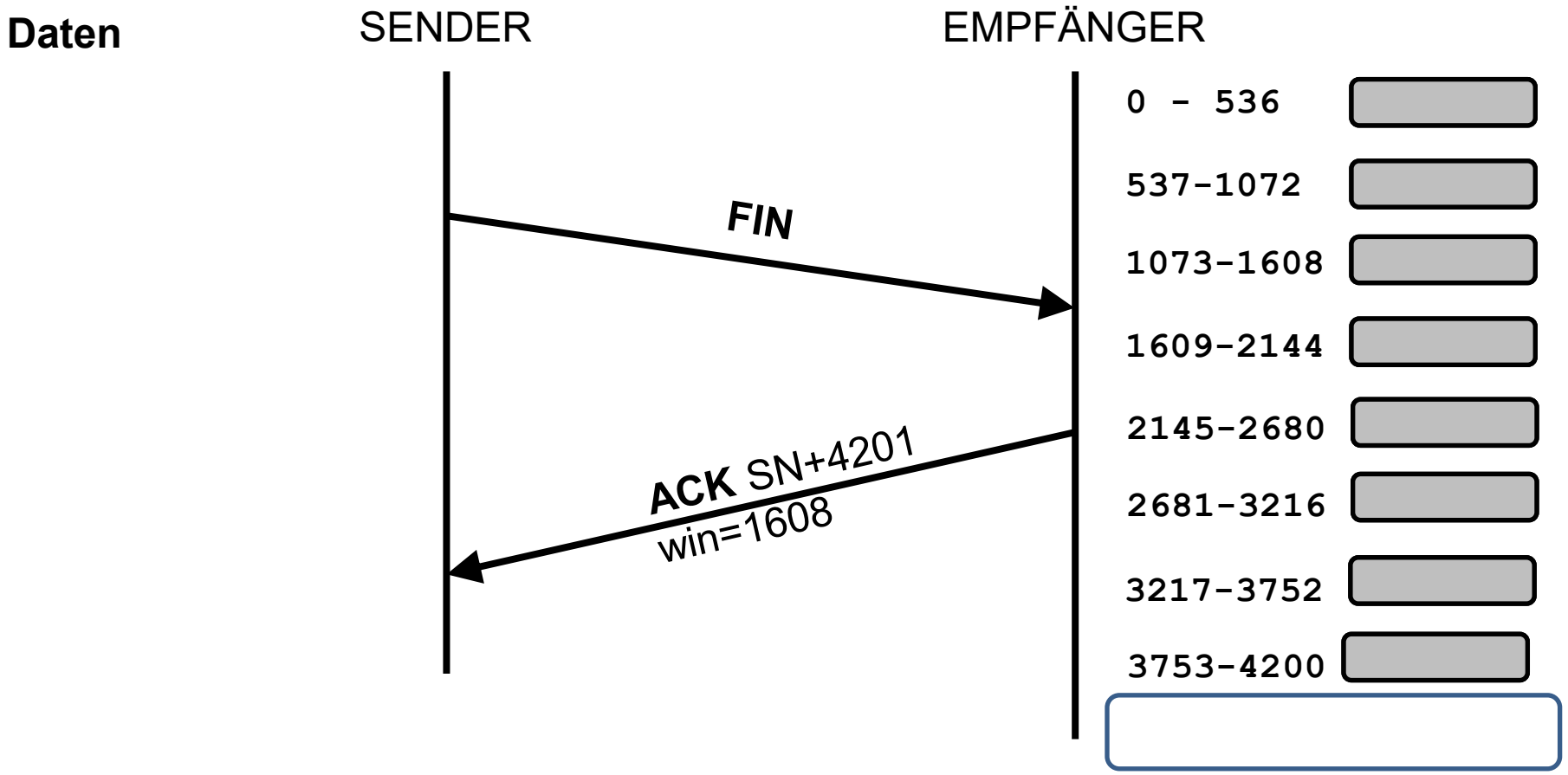
TCP: **Handshake & Sliding Windows** 6/8



TCP: **Handshake & Sliding Windows** 7/8



TCP: **Handshake & Sliding Windows** 8/8



Portnummern

- 0-1023: Well Known
- 1024-49151: Registered Ports
- 49152-65535: dynamische Ports

Port	Service
21	File Transfer Protocol (FTP)
22	Secure SHell (SSH)
23	Telnet
25	Simple Mail Transfer Protocol (SMTP)
53	Domain Name System (DNS)
80	Hyper Text Transfer Protocol (HTTP)
161	Simple Network Management Protocol (SNMP)
443	HTTP Secure (HTTPS)

Socket Verbindungen

- Kommt von Sockel oder Steckverbindung
- Ein Socket ist eine Standardschnittstelle zw.
  - Netzwerk
  - Anwendungssoftware
- Es gibt immer 2 Seiten:
  - Server
  - Client(s)
- Die Verbindung braucht eine **IP** und einen **Port**

Socket Kommunikation

- Server

1. Server Socket erstellen
2. Binden des Sockets an einen **Port**
3. Auf Anfragen warten
4. Bearbeiten der Anfrage
5. Schließen des Sockets-Port wird frei

- Client

1. Client Socket erstellen
2. Binden des Sockets mit **IP** und **Port** am Server
3. Senden und Empfangen
4. Schließen des Sockets nach Kommunikation

Socket Kommunikation | **Test-Server**

```
package test.webserver;

import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;

public class Server {

    public static void main(String[] args){
        Server unserServer = new Server(9000);
    }

    private ServerSocket meinServerSocket = null;
    private Socket meinClientSocket = null;

    public Server(int meinPort){
        try {

            this.meinServerSocket =
                        new ServerSocket(meinPort);
            System.out.println(
                        "Server hört nun auf Port " + meinPort);

            do {

                this.meinClientSocket =
                        this.meinServerSocket.accept();

                BufferedReader eingabeBuffer = new BufferedReader(
                        new InputStreamReader(
                                this.meinClientSocket.getInputStream()
                        )
                );

                String eingabe;
                StringBuffer sb = new StringBuffer();

                while((eingabe = eingabeBuffer.readLine()) != null){
                    if(eingabe.equalsIgnoreCase(""))
                        break;
                    sb.append(eingabe+System.lineSeparator());
                }

                PrintWriter out =
                        new PrintWriter(
                                this.meinClientSocket.getOutputStream());
                out.println("HTTP/1.1 200 OK");
                out.println("Content-Type: text/html");
                out.println("\r\n");
                String s = sb.toString();
                System.out.println(s);
                out.println("<pre> "+s+" </pre>");
                out.flush();
                out.close();

                this.meinClientSocket.close();
            }while(true);
```

Importe und Klassendefinition

Einstiegspunkt
Ein Server mit Port 9000

Notwendige Sockel werden definiert

"Konstruktor" der Klasse wird beim Initialisieren ausgeführt

***1**. Socket erstellen*
***2**. Binden an Port*

***3**. Auf Anfrage warten*

***4**. Bearbeiten der Anfrage*

Gepufferte Eingabe einrichten

So lange Zeilen lesen bis eine leere Zeile folgt …

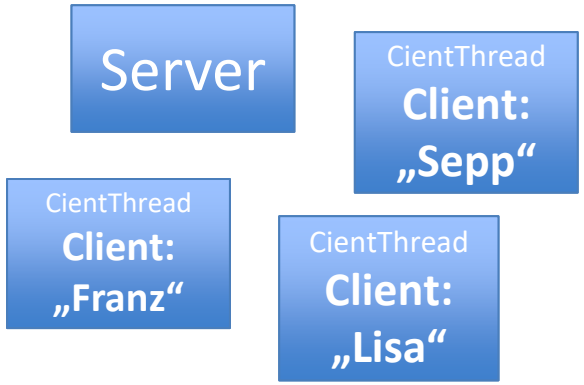
Eine Ausgabe einrichten

Eine HTTP Antort konstruieren, in der das Gelesene zurückgegeben wird

***5**. Schließen des Client Sockets*

Socket Kommunikation | Chat-**Server**

```
└──chat
    ├──client
    │       Client.java
    │
    └──server
            ClientThread.java
            Server.java
```



Hauptfunktion **main()**

1. Socket wird erstellt und auf Port gebunden
   `ServerSocket(`<Port>**`)`**
2. In einer Schleife werden eingehende Verbindungen akzeptiert
   `accept()`
3. Der Socket vom Client wird empfangen
   `Socket`
   und an die Klasse ClientThread weitergereicht, wo Eingabe und Ausgabe am Socket bearbeitet werden.

- **Server.java** verwaltet Clients
- **ClientThread.java** ist die "parallele" Verbindung zu jedem Client
- Hat einen Pool **an ClientThreads** (10 in diesem Beispiel)
- **Hört auf einen Port** (>1024)

SERVER

Socket Kommunikation | Chat-**Client**

```
└──chat
    ├──client
    │       Client.java
    │
    └──server
            ClientThread.java
            Server.java
```

Client

1. Der Client baut zum Server einen **`Socket(`**<IP>,<Port>**`)`** auf
2. Der Client bekommt Ein- und Ausgabeströme zur Verfügung:
   `getInputStream()`
   `getOutputStream()`

- **Client.java** verbindet sich mit dem Server
- Die Verbindungsdaten sind die **Server IP und der Port**, auf den der Server hört
- **Anmerkung:** Sie können auch gleichgut **"telnet"** verwenden

CLIENT

Socket Kommunikation | Chat-Server | **Quellcode**

```
package chat.server;

import java.io.*;
import java.net.*;

public class Server {
    // Default Port 8888
    static int PORT = 8888;

    // Socket for clients
    static Socket socketClient = null;
    // Socket for server
    static ServerSocket socketServer = null;
    // This chat server can accept up to 10 clients' connections
    static ClientThread clientthreadPool[] =
                                        new ClientThread[10];

    ..........
}
```

Platz für 10 Chatter:

SERVER

Socket Kommunikation | Chat-Server | **Quellcode**

```
public static void main(String args[]) {
    // Numbers of Threads
    int iThreadNumber = 0;

    // Default port
    int iPort = Server.PORT;

    // Given port from commandline
    if(args.length == 1){
        iPort = Integer.parseInt(args[0]);
    }

    // Try to open a server socket on port port_number (default 8888)
    // Note: Ports less than 1023 can only be defined by privileged users
    try {
        Server.socketServer = new ServerSocket(iPort);
    }// try
    catch (IOException e) {
        System.out.println("Error: Open server socket on port " + iPort);
        System.out.println(e);
        System.exit(1);
    }
```

1. Socket wird erstellt und auf Port gebunden

SERVER

Socket Kommunikation | Chat-Server | **Quellcode**

```
while (true) {
     try {
          Server.socketClient = Server.socketServer.accept();
          iThreadNumber++;

          ClientThread clientthreadTmp = new ClientThread(
                                        "Client-" + iThreadNumber,Server.socketClient, Server.clientthreadPool);

          if(Server.add2Pool(clientthreadTmp))
                    clientthreadTmp.start();
               else
          System.out.println("No slots free for new client!");

     }
     catch (IOException e) {
          System.out.println(e);
     }
}
```

2. In einer Schleife werden eingehende Verbindungen akzeptiert

```
public static synchronized boolean add2Pool(ClientThread clientthreadNew){
     boolean bInPool = false;
     for(int i=0; i<Server.clientthreadPool.length;i++){
          if(Server.clientthreadPool[i]==null){
               Server.clientthreadPool[i] = clientthreadNew;
               bInPool = true;
               break;
          }
     }
     return bInPool;
}
```

3. Der Socket vom Client wird empfangen und an die Klasse ClientThread weitergereicht, wo Eingabe und Ausgabe am Socket bearbeitet werden.

   Der ClientThread wird dem Pool hinzugefügt

SERVER

Socket Kommunikation | Client-Thread | **Quellcode**

```
class ClientThread extends Thread {
     // Read inputs
     private BufferedReader readerInput = null;
     // Output stream
     private PrintStream printstreamOut = null;
     // The client's socket
     private Socket socketClient = null;
     // The pool of client threads
     private ClientThread clientthreadPool[];
     // Thread's name
     private String sThreadName = null;
     // User's name
     private String sUserName = null;

     public ClientThread(
               String sThreadName,
               Socket clientSocket,
               ClientThread[] clientthreadpool) {

          this.sThreadName = sThreadName;
          this.socketClient = clientSocket;
          this.clientthreadPool = clientthreadpool;

     }
```

```
public synchronized void run() {
     // Line of text which is typed in
     String line = null;
     try {
          // Establish input reader
          this.readerInput =
               new BufferedReader(
                    new InputStreamReader(
                         this.socketClient.getInputStream()));
          // Establish output stream
          this.printstreamOut =
               new PrintStream(
                    this.socketClient.getOutputStream());

          // Greet and get name of user by input
          this.printstreamOut.println("Chat by Mathias Knoll (C2009)");
          this.printstreamOut.println("Enter your name.");
          this.sUserName = this.readerInput.readLine();
          this.sThreadName = this.sThreadName + "-" + this.sUserName;
          // Welcome user with his name
          this.printstreamOut.println("Hello " + this.sUserName);
          System.out.println("New user " + sUserName + ".");
```

SERVER

Socket Kommunikation | Client-Thread | **Quellcode**

```
            // Deliver message of new user to all other users
            for (int i = 0; i < this.clientthreadPool.length; i++){
                  // If thread is there and not this one...
                  if (
                        this.clientthreadPool[i] != null &&
                        this.clientthreadPool[i] != this){
                              this.clientthreadPool[i].printstreamOut.println(this.sUserName+"!");
                  }
            }
            // Start threads endless loop (until user quits)
            while (true) {
                  // Get input
                  line = readerInput.readLine();
                  // Send input to all other clients in pool
                  for (int i = 0; i < this.clientthreadPool.length; i++){
                        if (this.clientthreadPool[i] != null){
                              this.clientthreadPool[i].printstreamOut.println(
                                                      "<" + this.sUserName + "> " + line);
                        }
                  }
            }
            // Leaving message to this client
            this.printstreamOut.println("*** Bye " + this.sUserName + " ***");
                  } catch (IOException e) {
                        // Error
                  }
}
```

SERVER

Socket Kommunikation | Chat-Client | **Quellcode**

```
package chat.client;

import java.io.*;
import java.net.*;

public class Client extends Thread {
    // Default values
    static String HOST = "localhost";
    static int PORT = 8888;
    private String sHost = Client.HOST;
    private int iPort = Client.PORT;
    // Socket
    private Socket socketClient = null;
    // Input reader
    private BufferedReader readerInput = null;
    // Output stream
    private PrintStream streamOut = null;
    // Reader for input line
    private BufferedReader readerLine = null;
    // Flag for stopping thread
    private boolean bStopped = false;
```

```
public static void main(String[] args) {
    // Host and port have to be delivered
    if(args.length != 2){
        System.out.println("Please deliver arguments
        for host and port!");
    }else{
        // Initialize Client!
        Client client = new Client(
                args[0],Integer.parseInt(args[1]));
        client.initialize();
    }
}

// Constructor
public Client(String sHost, int iPort){
    // Get host
    this.sHost = sHost;
    // Get port
    this.iPort = iPort;

}
```

CLIENT

Socket Kommunikation | Chat-Client | **Quellcode**

```
private void initialize(){
     // Outgoing line of text
     String sLine = null;
     try {
          // Establish socket to server
          this.socketClient = new Socket(this.sHost, this.iPort);
          // Reading input from system.in
          this.readerLine =
               new BufferedReader(
                    new InputStreamReader(
                         System.in));
          // Stream out
          this.streamOut = new PrintStream(this.socketClient.getOutputStream());
          // Read input from socket
          this.readerInput =
               new BufferedReader(
                    new InputStreamReader(
                              this.socketClient.getInputStream()));
     } catch (UnknownHostException e) {
          System.err.println("Don't know about host " + Client.HOST);
     } catch (IOException e) {
          System.err.println("Couldn't get I/O for the connection to the host " + Client.HOST);
     }
```

CLIENT

Socket Kommunikation | Chat-Client | **Quellcode**

```
if (
     this.socketClient != null &&
     this.streamOut != null &&
     this.readerInput != null) {

     try {
          // Start thread
          this.start();
          // Send read line
          while (!this.bStopped) {

               sLine = this.readerLine.readLine();

               // Deliver stream to client thread
               this.streamOut.println(sLine);

          }
          this.streamOut.close();
          this.readerInput.close();
          this.socketClient.close();

     } catch (IOException e) {
          System.err.println("IOException: " + e);
     }
}
}
```

```
public void run() {

     // Incoming line of text
     String line = null;

     try {
          while (
              (line = this.readerInput.readLine()) != null
          ) {

               System.out.println(line);

               // If message contains a 'bye' message
               // leave thread
               if (line.indexOf("*** Bye") != -1)
                    break;
          }
          bStopped = true;
     } catch (IOException e) {
          System.out.println(
            "Error: Server connection failed! Exiting!");
          }
     }
```

CLIENT

DIHSÜD

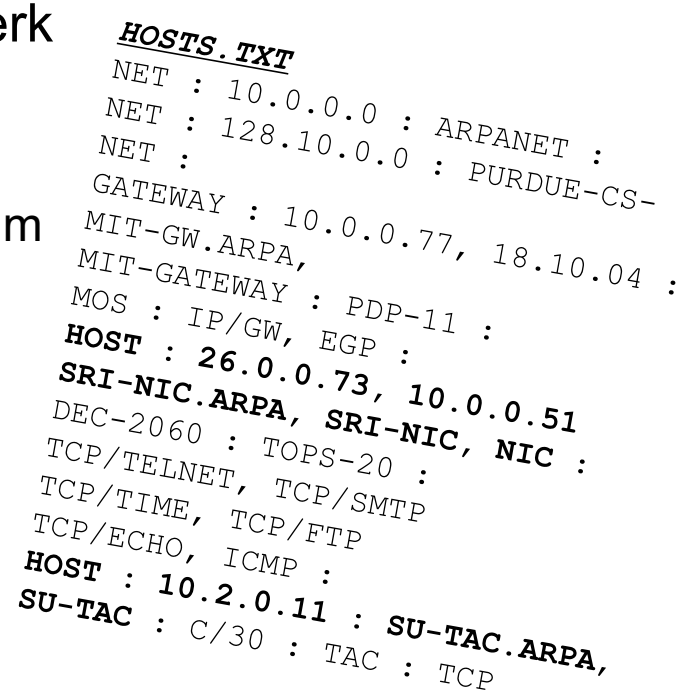
FH JOANNEUM
University of Applied Sciences

Anwendungsorientierte Schichten
OSI 5,6,7

Domain Name System (DNS) | 1983

Vor DNS

- Namensauflösung per **simpler Textdatei**, welche auf alle Hosts im Netzwerk verteilt wurde
- Probleme:
  Rasanter Wachstum des ARPAnet
  Namenskonflikte durch/und inkonsistente Dateien

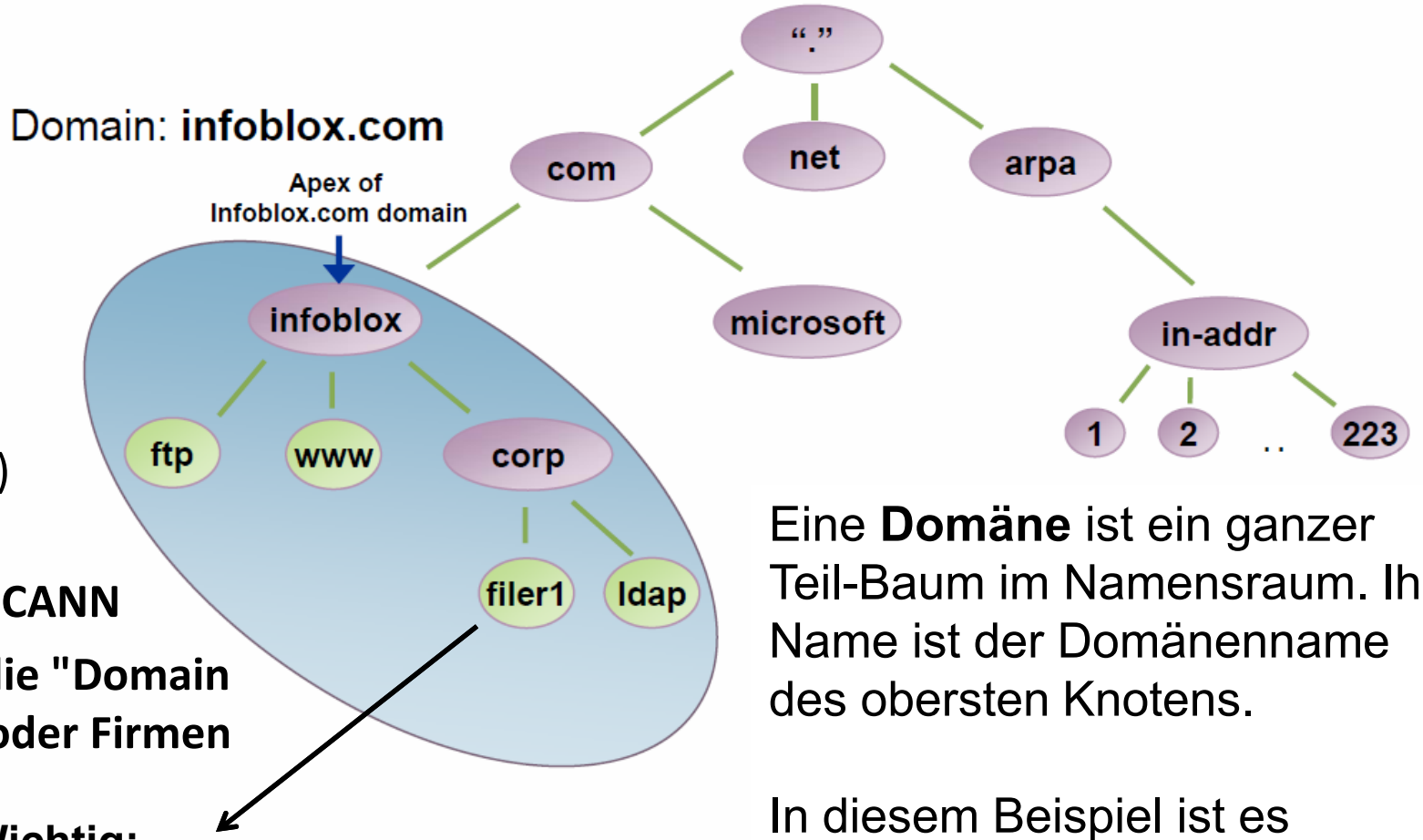


Nach DNS

- Schnelle Suche (**Abfragen**)
- Replizierte Inhalte (**Ausfallsicherheit**)
- **Verteilte Kontrolle** (Zonen)
- Anfragen gehen über **UDP Port 53** mit **Maximallänge von 512Byte**
- **Zonentransfers** und lange Abfragen/Antworten gehen über **TCP Port 53**

DNS | **Aufbau**

- **Hierarchisch** aufgebaut
- **Skalierbar**
- **Dezentrale** Organisation
  - Eine Organisationseinheit (OE) verwaltet eine **Domäne**
  - **Die OE für Rootserver ist die ICANN**
  - Für **TLD delegiert die ICANN die "Domain Registry" an Organisationen oder Firmen**
  - **Pro Zone kann es mehr als einen Namensserver geben**



Wichtig:
FQDN – Fully Qualified Domain Name
filer1.corp.infoblox.com.

Eine **Domäne** ist ein ganzer Teil-Baum im Namensraum. Ihr Name ist der Domänenname des obersten Knotens.

In diesem Beispiel ist es **infoblox.com.**

DNS | **Funktion & Werkzeuge**

Vorwärts-Auflösung (Forward Mapping)

- Name zu Adresse
- Bsp: **`dakar.fh-joanneum.at` → `62.218.221.66`**

Rückwärts-Auflösung (Reverse Mapping)

- Adresse zu Name
- Bsp: **`66.221.218.62.in-addr.arpa` → `dakar.fh-joanneum.at`**
- ***Achtung***: Man beachte, dass es fürs Reverse Mapping eine spezielle Hierarchie gibt, in der die IP umgekehrt inkorporiert ist! An deren Ende steht statt **in-addr.arpa**!

Klassisches Werkzeug: `nslookup`

host

- **# host -t ns at.**
- Findet Namensserver Einträge für die TLD "at"

nslookup

- **# nslookup orf.at**
- Namensauflösung

dig

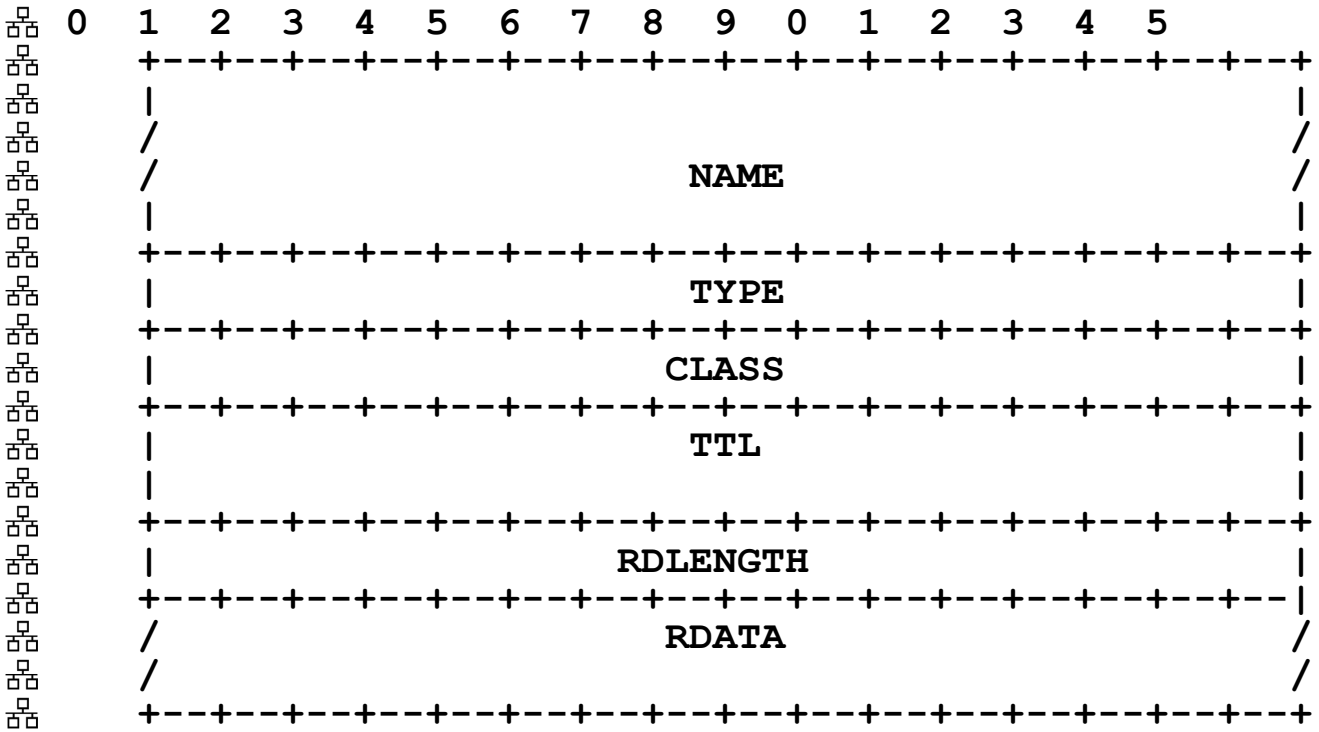
- Namensauflösung mit mehr Optionen

DNS | **Wie funktioniert es?**

1. **Anfrage am PC über Resolver**
2. **Ein DNS Server wird gefragt (rekursiv)**
3. **Dieser fragt einen Root Server**
4. **Dieser gibt einen TLD Server zurück**
5. **Dieser gibt den Server einer OE zurück**
6. **Dieser gibt die Adresse zurück**
7. **Die Adresse wird an den PC zurückgegeben.**

Resolver =
Programm oder Bibliothek, die für Namensauflösungen genutzt wird!

DNS | **RFC1035**

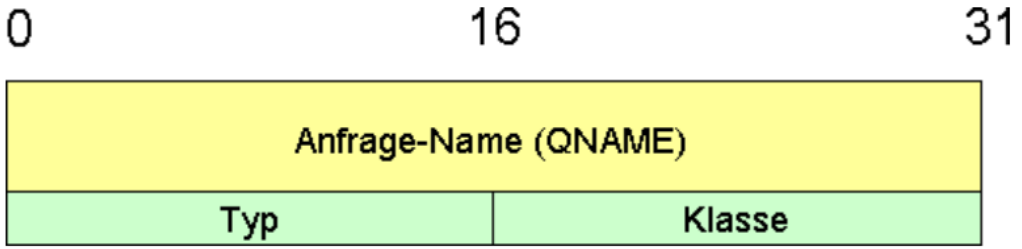


Ein Datensatz besteht aus:

- **Name** (optional)
  Domänenname zu dem der Datensatz gehört
- **Gültigkeit** (optional)
  Die TTL in Sekunden
- **Klasse** (optional)
  IN für Internet- andere werden nicht mehr verwendet
- **Typ**
  Die Typen werden später im Detail durchgenommen
- **Ressource Daten**
  Beschreiben den Datensatz

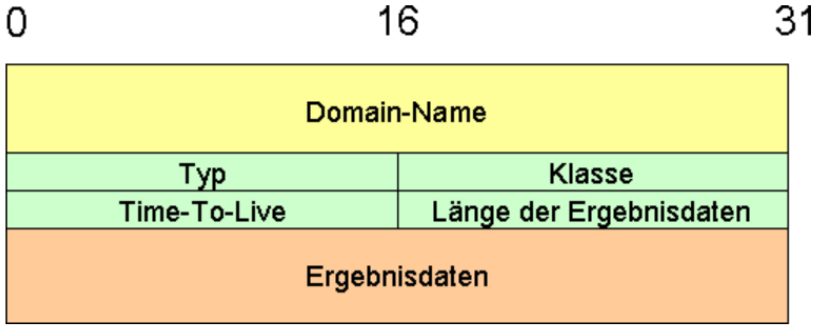
DNS | **Nachrichtenformat: Anfrage (Request)**

```
> Frame 26: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
> Ethernet II, Src: AsustekC_da:58:ea (08:60:6e:da:58:ea), Dst: CiscoInc_ff:fd:2c (00:08:e3:ff:fd:2c)
> Internet Protocol Version 4, Src: 10.52.37.222, Dst: 10.65.1.2
> User Datagram Protocol, Src Port: 63826, Dst Port: 53
v Domain Name System (query)
    [Response In: 27]
    Transaction ID: 0x0004
  v Flags: 0x0100 Standard query
      0... .... .... .... = Response: Message is a query
      .000 0... .... .... = Opcode: Standard query (0)
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... .0.. .... = Z: reserved (0)
      .... .... ...0 .... = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    v fh-joanneum.at: type A, class IN
        Name: fh-joanneum.at
        [Name Length: 14]
        [Label Count: 2]
        Type: A (Host Address) (1)
        Class: IN (0x0001)
```



DNS | **Nachrichtenformat: Antwort (Response)**

```
> Frame 27: 90 bytes on wire (720 bits), 90 bytes captured (720 bits) on interface 0
> Ethernet II, Src: CiscoInc_ff:fd:2c (00:08:e3:ff:fd:2c), Dst: AsustekC_da:58:ea (08:60:6e:da:58:ea)
> Internet Protocol Version 4, Src: 10.65.1.2, Dst: 10.52.37.222
> User Datagram Protocol, Src Port: 53, Dst Port: 63826
v Domain Name System (response)
    [Request In: 26]
    [Time: 0.000298000 seconds]
    Transaction ID: 0x0004
  v Flags: 0x8580 Standard query response, No error
      1... .... .... .... = Response: Message is a response
      .000 0... .... .... = Opcode: Standard query (0)
      .... .1.. .... .... = Authoritative: Server is an authority for domain
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... 1... .... = Recursion available: Server can do recursive queries
      .... .... .0.. .... = Z: reserved (0)
      .... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
      .... .... ...0 .... = Non-authenticated data: Unacceptable
      .... .... .... 0000 = Reply code: No error (0)
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    v fh-joanneum.at: type A, class IN
        Name: fh-joanneum.at
        [Name Length: 14]
        [Label Count: 2]
        Type: A (Host Address) (1)
        Class: IN (0x0001)
  v Answers
    v fh-joanneum.at: type A, class IN, addr 91.118.154.85
        Name: fh-joanneum.at
        Type: A (Host Address) (1)
        Class: IN (0x0001)
        Time to live: 1800
        Data length: 4
        Address: 91.118.154.85
```



DNS | **Resource Records 1**

A

```
name        IN      A       adresse
www         IN      A       192.168.1.1
```

CNAME

```
name        IN      CNAME   ziel
ftp         IN      CNAME   www
```

NS

```
domain      IN      NS      server
@           IN      NS      ns.servertec.
servertec.    IN      NS      ns.servertec.
```

MX

```
domain      IN      MX      priority server
@           IN      MX      10 mail.servertec.
servertec.    IN      MX      10 mail.servertec.
```

DNS | **Resource Records 2**

PTR *(Reverse Lookup! *IP in umgekehrter Reihenfolge!)*

```
ip*         IN      PTR     name
1           IN      PTR     www.servertec.
```

SOA

```
zone        IN      SOA     primary
                            mailaddresse
                            seriennummer
                            refresh retry expire
                            TTL
@           IN      SOA     ns.servertec.
                            root.servertec.
                            2013020201
                            3600 1800 604800
                            600
```

Dynamic Host Configuration Protocol (DHCP) | 1996

Vor DHCP

- **Viel Arbeit** bei der Einrichtung!

- Schritt 1: Vergebene **IP noch in Verwendung?**
- Schritt 2: **PING** zum Host
- Schritt 3: **Warten**
- Schritt 4: **Wenn keine Rückmeldung** kommt kann man die **IP Adresse vergeben**

Frage: Was ist, **wenn ICMP deaktiviert ist?**

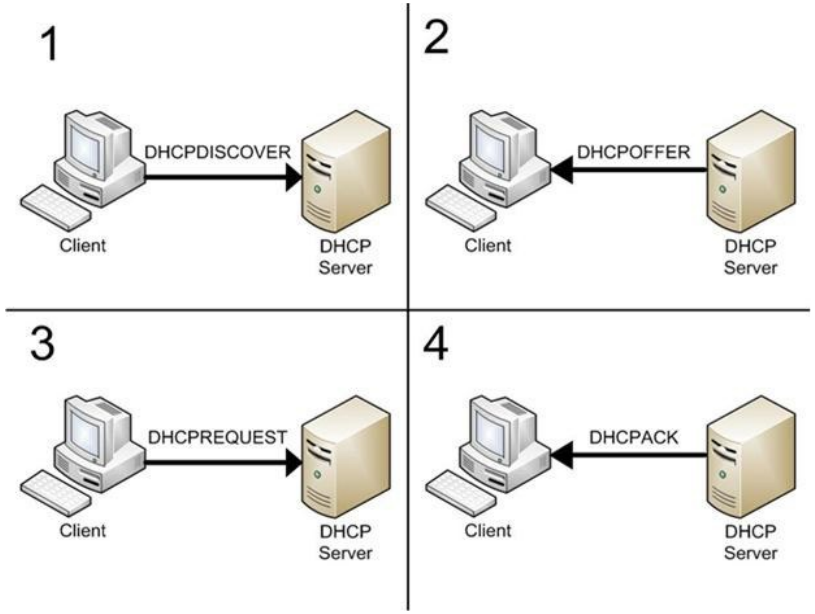
Nach DHCP

- **Automatische Zuordnung** von IP Adressen
- Wireless LAN Acccess Points möglich!
- Details zur IP Vergabe sind konfigurierbar.
- Zusatzinfos zur Netzwerk-Infrastruktur werden mitgeliefert (Gateway, DNS Server, usw.)
- Neue Infrastrukturen können über den Server verteilt werden (DHCPFORCERENEW)

DHCP | **Dienst**

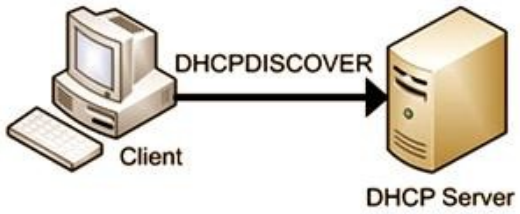
- Protokoll

  IP Vergabe "DORA":
  DISCOVER, OFFER, REQUEST, ACK.



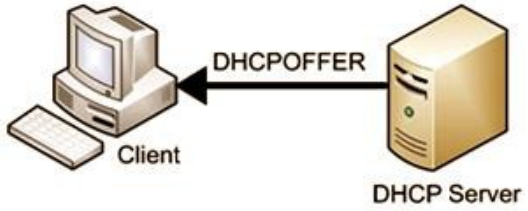
- Normalerweise konkurrieren sich 2 DHCP Server in einem Netzwerk (**Rogue DHCP**)
- Konfiguriert man beide jedoch entsprechend sind jedoch mehr als einer möglich (**Server Conflict Detection**)
- Ohne DHCP: Sebstkonfiguration der Clients
  - zeroconf (zero configuration networking)
  - APIPA (Automatic Private IP Addressing) oder Auto-IP unter Windows
  - Avahi bei Linux oder
  - Bonjour bei Apple

DHCP | **Protokoll**



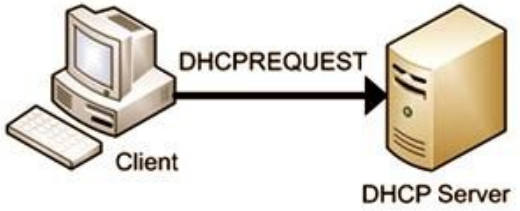
DHCPDISCOVER

- Broadcast
- Quelladresse: 0.0.0.0
- Zieladresse: 255.255.255.255
- Quell-MAC-Adresse



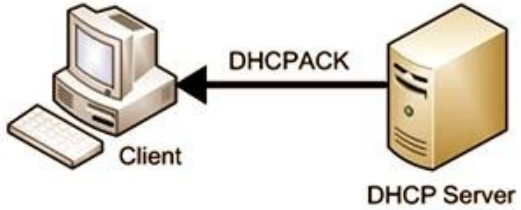
DHCPOFFER

- Eine freie IP Adresse wird an den Client gesendet
- Der Client wird an seiner MAC Adresse erkannt!



DHCPREQUEST

Der Client bestätigt die Verwendung der ihm zugewiesenen Adresse



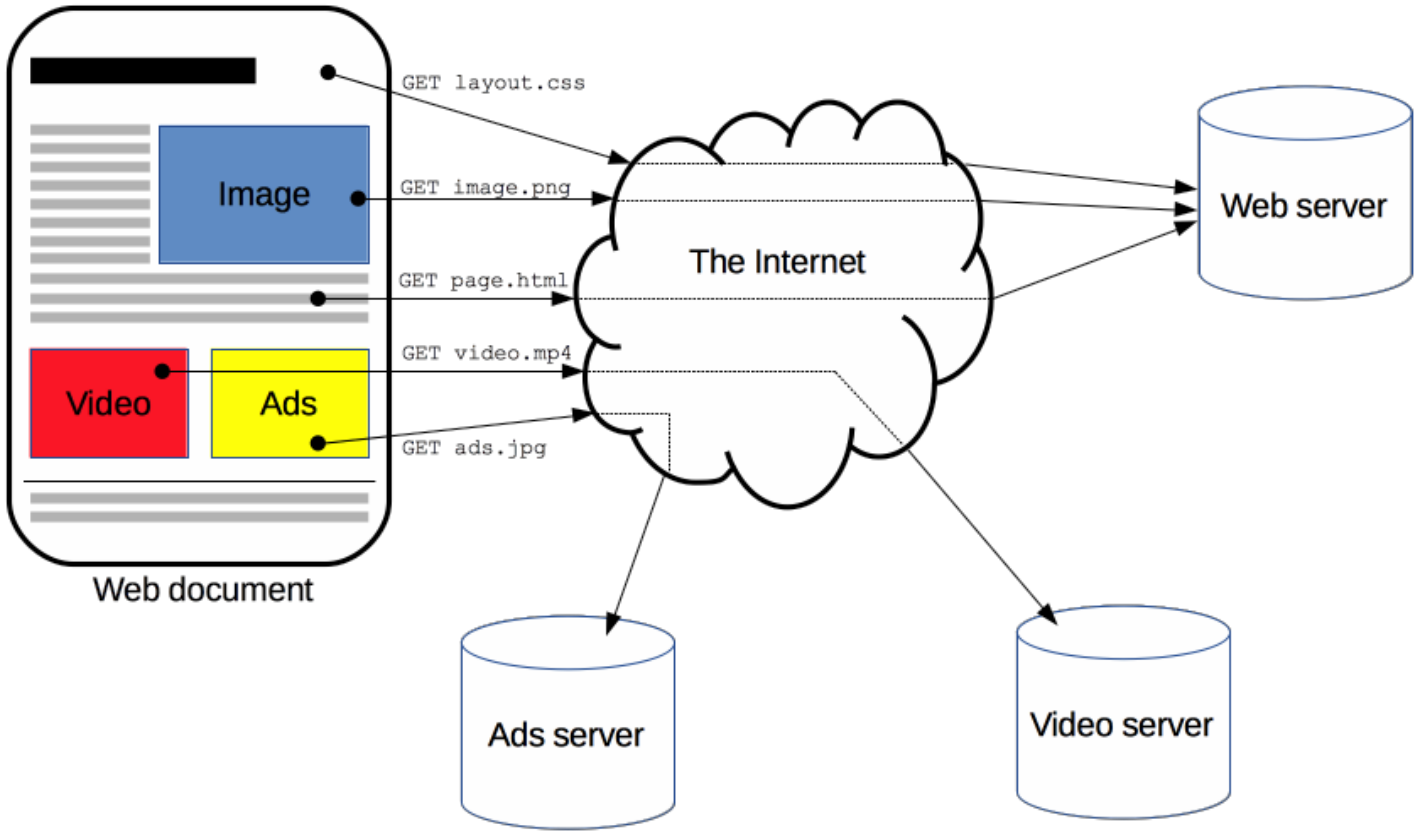
DHCPACK

Der Server bestätigt die erfolgreiche Zuweisung der Adresse

DHCPDECLINE
Client informiert Server, dass die IP schon benutzt. wird.
Der Server sperrt die IP für weiteres.
DHCPRELEASE
Client gibt seine IP frei!
Normalerweise beim Shut-Down verwendet.

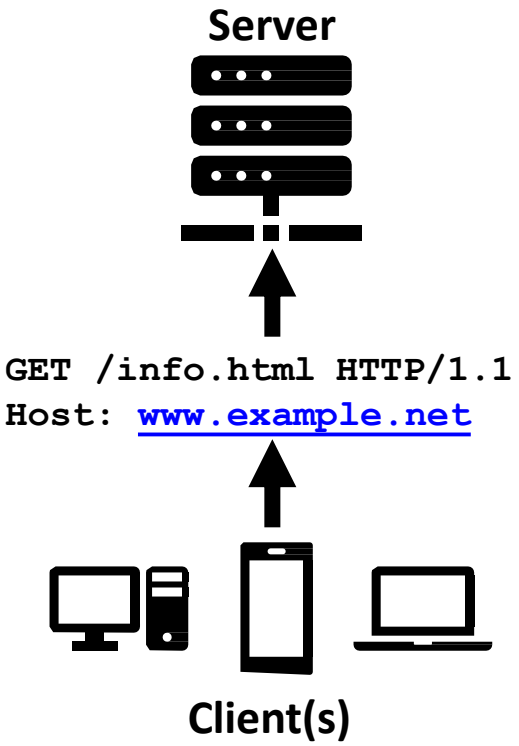
DHCPNAK
Der Server verweigert eine Client Anfrage
DHCPINFORM
Beliefert Clients mit weiteren Informationen (z.B.: über andere Server)
DHCPFORCERENEW
Pro-Aktiv an alle Clients vom Server gesendet, damit alle neue IPs anfordern.

Hyper Text Transfer Protocol (HTTP) | 1989



https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

HTTP | **Flow**

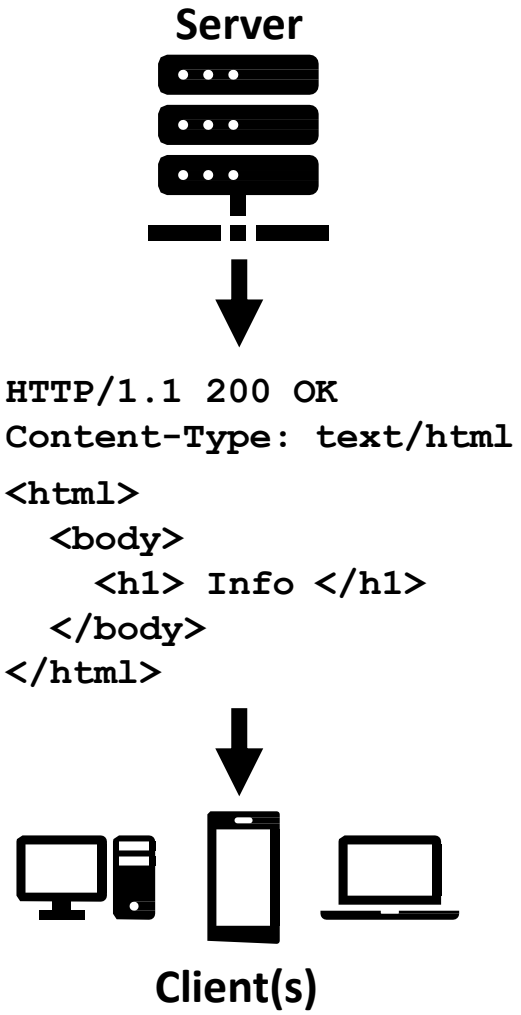


Client Request

- Browser auf irgendeinem Gerät (PC, Laptop, Smartphone)
- **sendet** eine (HTTP-) **Anfrage**
- an den (Web-) **Server**.

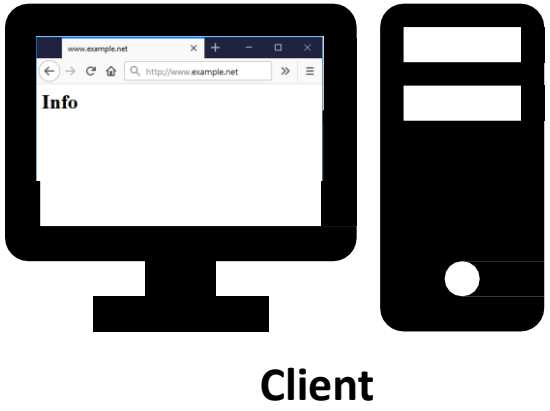
Server Response

- Server (Virtuelle Maschine, physisch in einem Rack)
- **schickt** eine (HTTP-) **Antwort**
- mit **einem (HTML) Dokument**
- an den (Web-) **Client**.

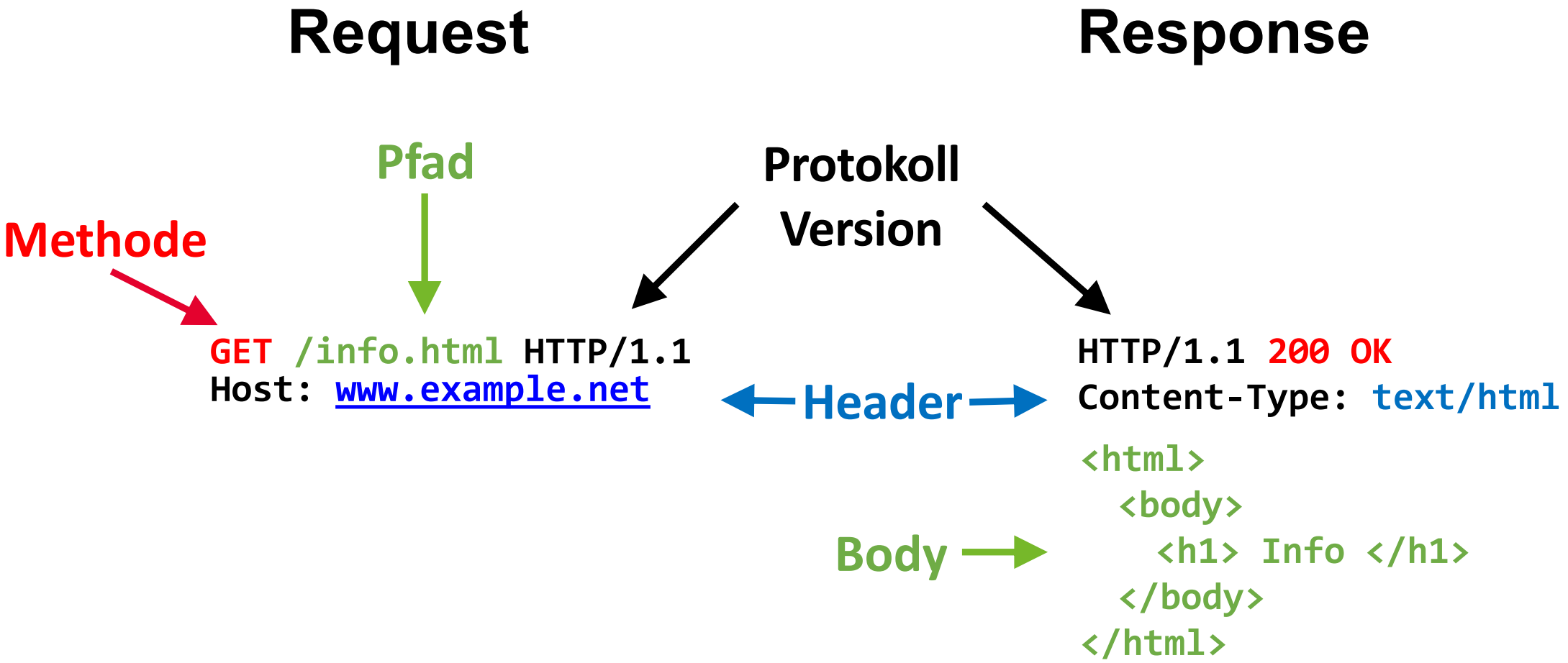


Client

- **rendert** das (HTML-) **Dokument am Bildschirm**



HTTP | **Protokoll**



HTTP | **Methoden**

"Sicher"
Keine Aktion am Server

- **GET** — Abrufen einer Ressource
- **HEAD** — Inspizieren der Ressourcen Header

Nachrichten mit "Body"
Daten zum Server senden

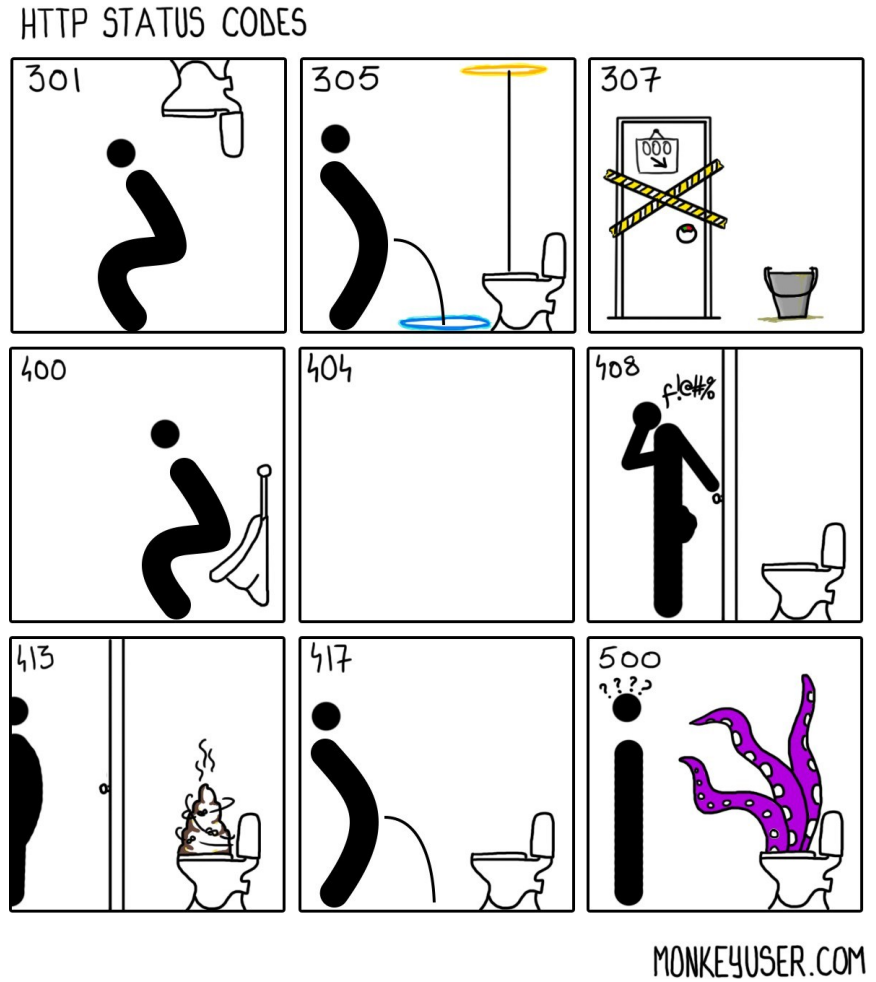
- **POST** — Senden von Daten zur Verarbeitung
- **PUT** — Ablegen von Daten am Server
- **PATCH** — Teilweise Modifikation von Ressourcen

- **TRACE** — Zurücksenden der erhaltenen Nachricht
- **OPTIONS** — Abrufen der Server Eigenschaften
- **DELETE** — Löschen einer Ressource

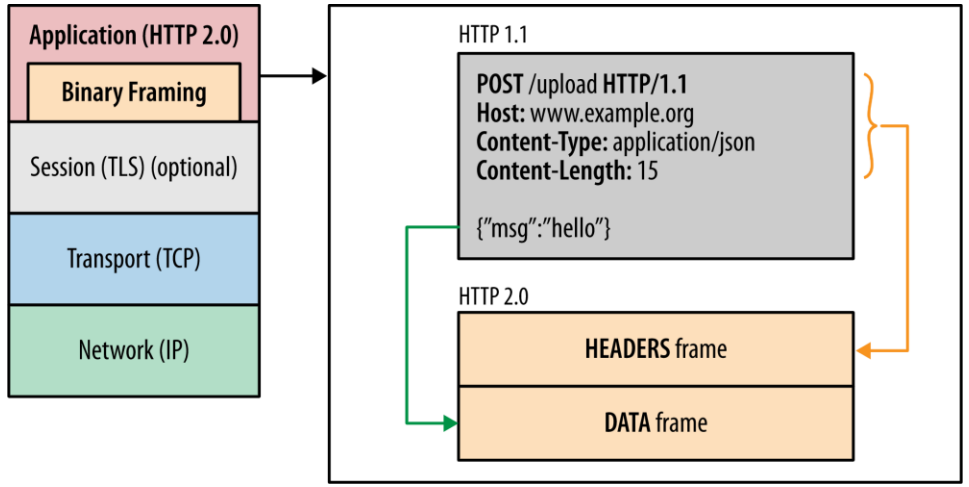
HTTP | **Status Codes**

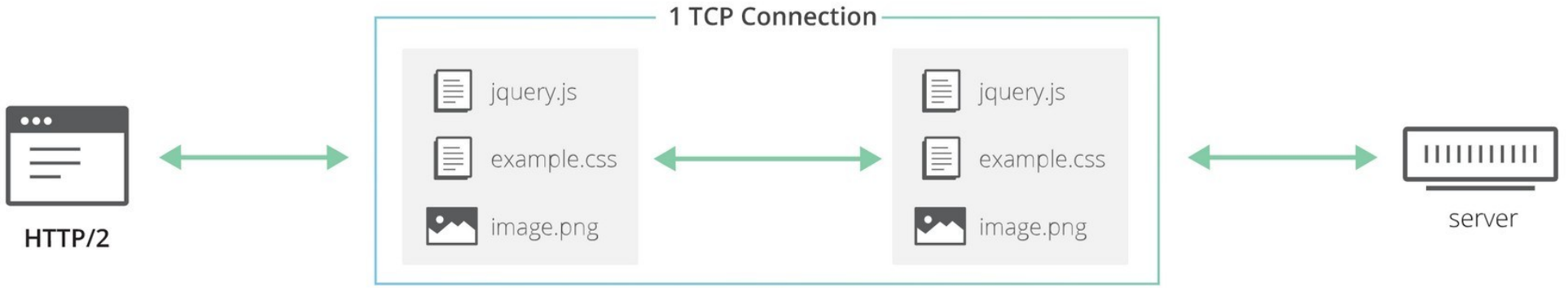
2xx Success	200	OK
3xx Redirection	301	Moved Permanently
	305	Use Proxy
	307	Temporary Redirect
4xx Client Error	400	Bad Request
	404	Not Found
	408	Request Timeout
	413	Payload Too Large
	417	Expectation Failed
5xx Server Error	500	Internal Server Error

*1xx Informational Response

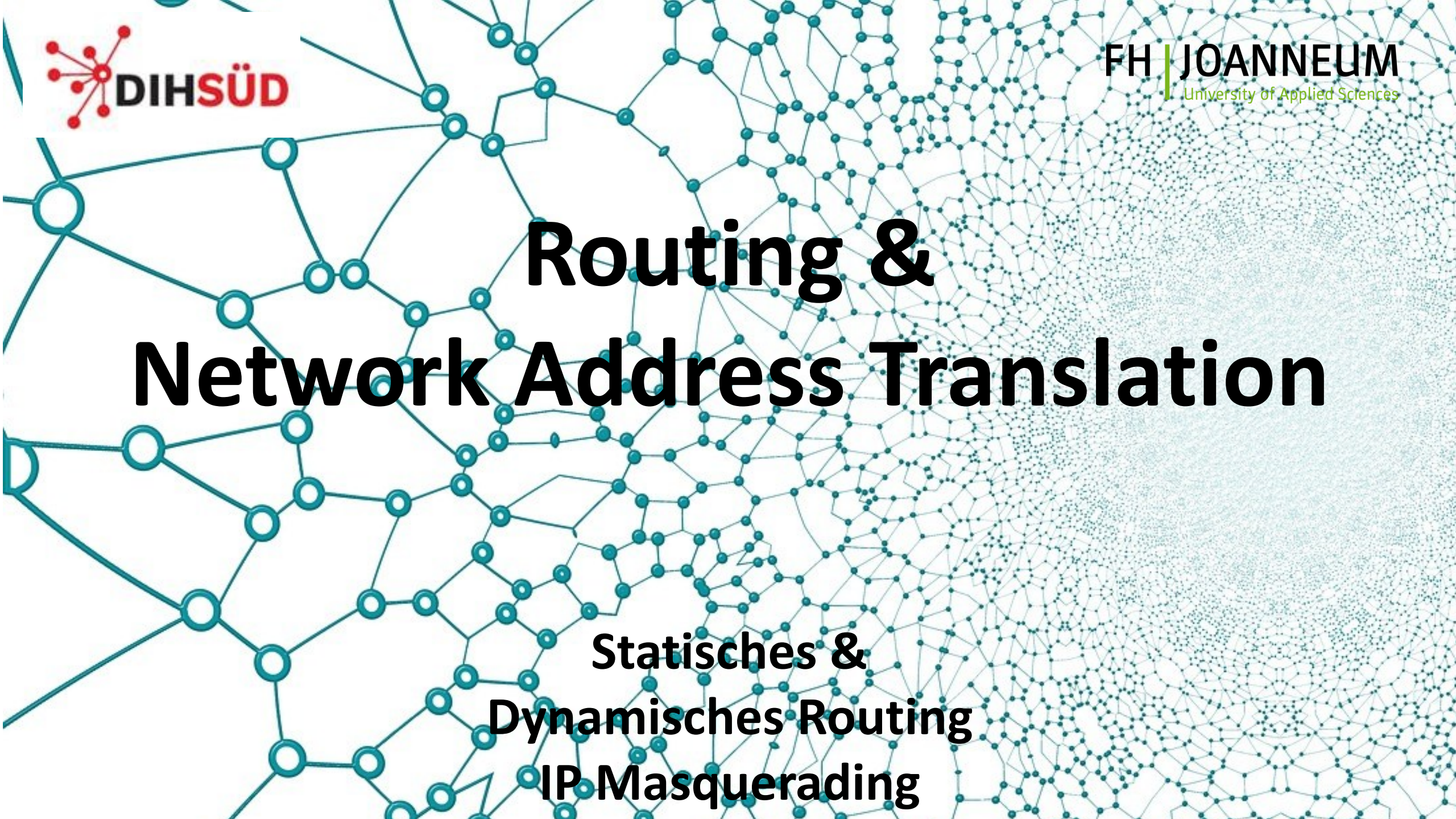


HTTP | **Version 2**





https://developers.google.com/web/fundamentals/performance/http2/



Routing & Network Address Translation

Statisches & Dynamisches Routing IP Masquerading

Routing | Allgemeines

Statisches Routing

- Routen werden **vom Netzwerkadministrator manuell** eingerichtet
- Routen können **nur manuell verändert** werden
- **Änderungen der Netzwerktopologie werden nicht automatisch erkannt** und das Routing adaptiert
- Muss **bei allen Routern und Hosts** gemacht werden
- Spezialfall: **Standardroute (Default Route) über den Gateway**
- Befehle: `ip route` (Linux)

Geroutete Protokolle

- Netzwerkprotokolle, die eine **Kommunikation über Netzwerkgrenzen ermöglichen**
- Adressierungsschema **mit Netzwerk und Host-Teil** notwendig
- Beispiele**: IPv4, IPv6**

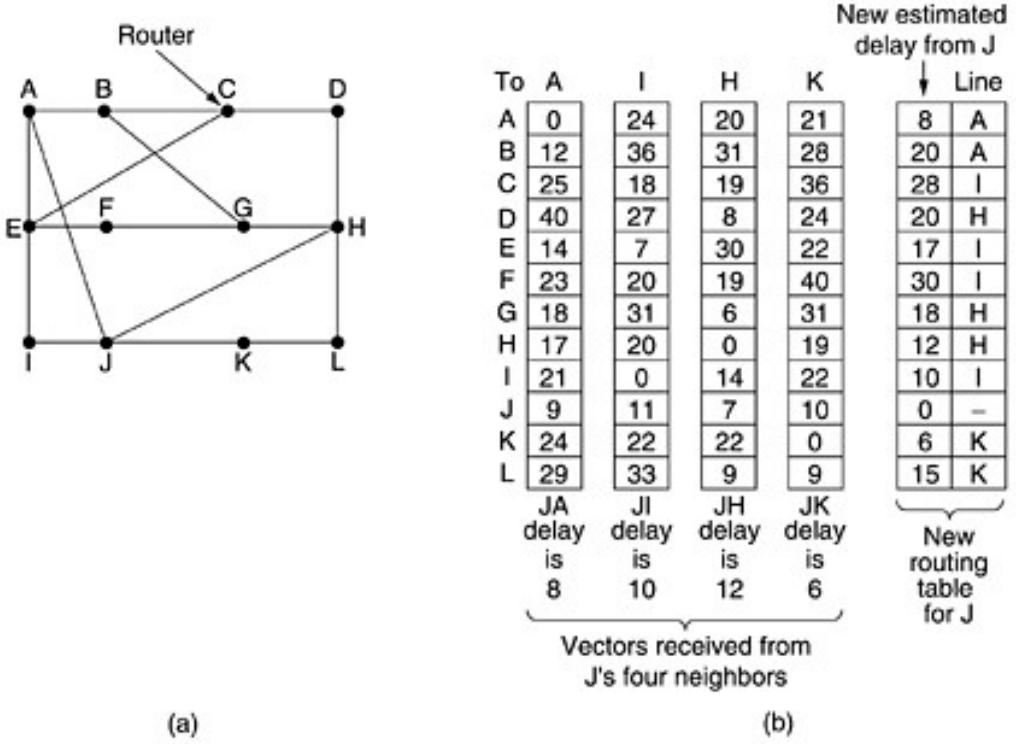
Routingprotokolle (Dynamisches Routing)

- Tauschen **Informationen zwischen Routern** aus
- Dienen zum **Finden der optimalen Route**
- **Dynamisches Routing** – Änderungen in der Topologie werden automatisch erkannt
- Beispiel: **Routing Information Protocol (RIP)**
- Verwenden von **Metriken** wie Hops*, Bandbreite, Zuverlässigkeit, Netzwerklast, Verzögerungen oder Kosten

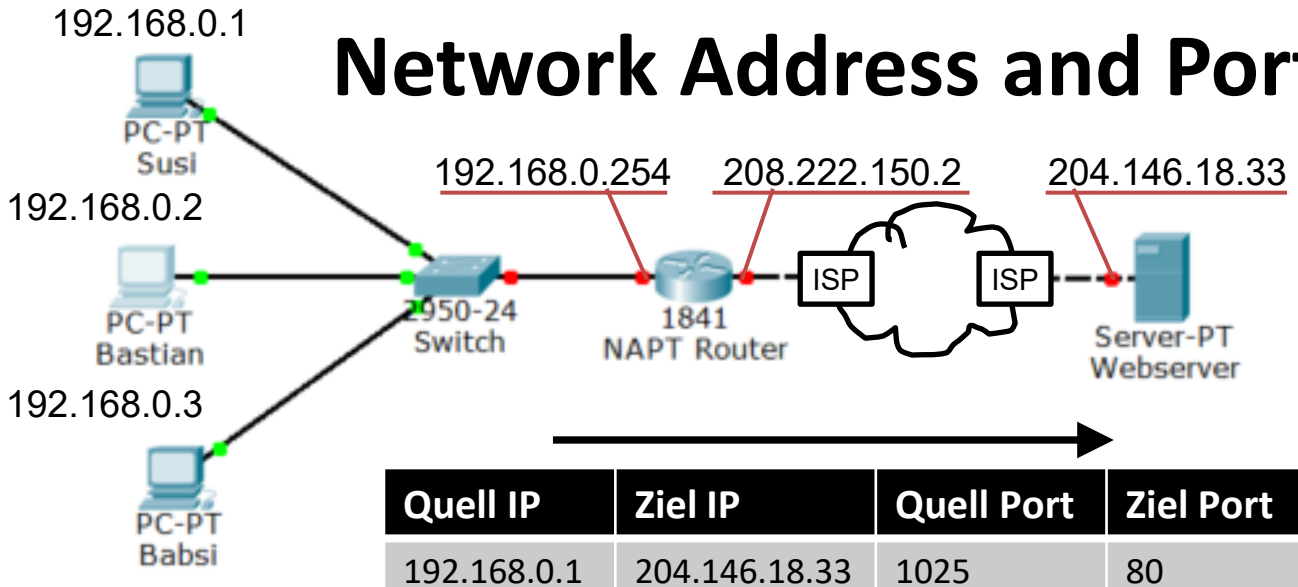
(* Ein Hop wird das Passieren von einem Netzwerksegment zum anderen genannt)

Funktionsweise von RIP

„Computer Networks“, Tanenbaum



Network Address and Port Translation (NAPT)



Quell IP	Ziel IP	Quell Port	Ziel Port
192.168.0.1	204.146.18.33	1025	80

Quell IP	Ziel IP	Quell Port	Ziel Port
208.222.150.2	204.146.18.33	55336	80

Adresse	192.168.0.1	208.222.150.2
Port	1025	55336

Masquerading Tabelle

Quell IP	Ziel IP	Quell Port	Ziel Port
204.146.18.33	192.168.0.1	80	1025

Quell IP	Ziel IP	Quell Port	Ziel Port
204.146.18.33	208.222.150.2	80	55336


DIHSÜD
FH | JOANNEUM
University of Applied Sciences
Danke für Ihre Aufmerksamkeit
Fortsetzung:
WebEngineering

DIH SÜD
Digital Innovation Hub

DIGITALISIERUNG IST EASY

Folgen Sie uns

für mehr Informationen zu Veranstaltungen, Digitalisierung und Innovation.

FH | JOANNEUM
University of Applied Sciences

INSTITUT Software Design und Security
Werk-VI-Straße 46
8605 Kapfenberg, AUSTRIA
T.: +43 316 5453 – 8374
E.: iit@fh-joanneum.at