

# TCP/IP

## Teil 1: Theoretische Grundlagen

Johannes Franken  
<jfranken@jfranken.de>

### Kursinhalt „Theoretische Grundlagen“

- [Kapitel 1: Der TCP/IP Protocol Stack](#)
  - Einführung in Protokolle und Protocol Stacks
  - Aufbau von TCP/IP
- [Kapitel 2: Link-, Network- und Transport- Layer](#)
  - Beschreibung der kernelnahen Layer
  - Netz-Hardware
- [Kapitel 3: Application Layer](#)
  - Das Domain Name System
  - Darstellung einiger Anwendungsprotokolle
- [Kapitel 4: Fachliteratur](#)
  - Fachliteratur
- [Kapitel 5: Diskussion](#)
  - Fragen/ Vorschläge zum 2.Teil/ Feedback

### Kapitel 1: Der TCP/IP Protocol Stack

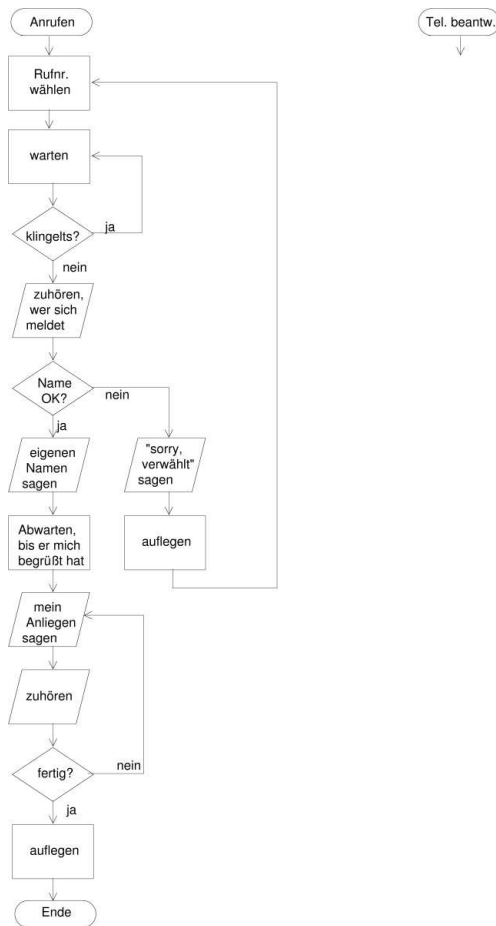
#### Wer oder was ist TCP/IP

TCP/IP bezeichnet einen *Protocol Stack*, der die *Protokolle* TCP<sub>\*</sub> und IP<sub>\*</sub> verwendet. Er

- entstand etwa 1968 aus dem ARPA<sub>\*</sub> net-Projekt des US Verteidigungsministeriums.
- wurde zum Aufbau des heutigen Internet verwendet.
- wird heute in vielen Firmen eingesetzt, z.B. anstelle von IPX, Netbeui oder SNA.
  - um eigene Dienste im Internet anzubieten
  - um fremde Dienste aus dem Internet im Firmennetz anzuwenden
  - weil es gut dokumentiert ist
  - weil es lizenzfrei eingesetzt werden darf
  - weil es offensichtlich hervorragend funktioniert

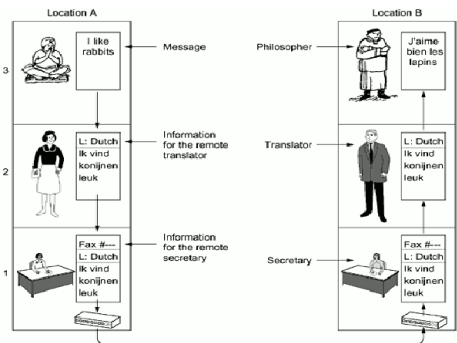
#### Beispiel Protokolle

Ein Protokoll ist die exakte Beschreibung eines Vorgangs.



## Protocol Stacks

Ein Protocol Stack ordnet Protokolle in Layer.



(Beispiel aus Tanenbaum, Computer Networks)

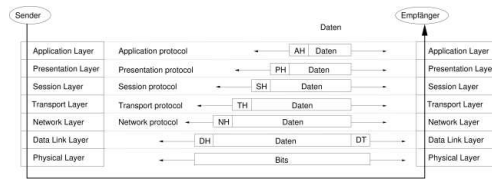
Ziel ist die Austauschbarkeit der Protokolle innerhalb eines Layer ohne Beeinflussung der übrigen Layer.

## Populäre Protocol Stacks

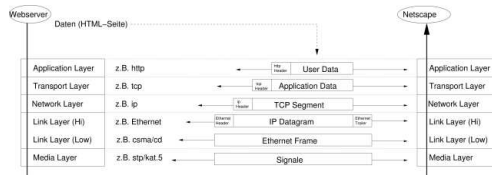
In Reihenfolge ihrer Entstehung:

1. SNA\* von IBM: sieben Layer, rechtlich geschützt
2. TCP/IP: etwa vier Layer (Grenzen verschwommen), Protokolle verschiedener Schichten voneinander abhängig, aber erfolgreich
3. IPX/SPX von Novell: vier Layer, Konzept von XNS\* übernommen
4. OSI\* von ISO\*: sieben Layer, frei und perfekt aber zu spät

# Layer im ISO/OSI Protocol Stack



# Layer im TCP/IP Protocol Stack



# Zoom into TCP/IP Protocol Stack

## Wichtige Erkenntnisse

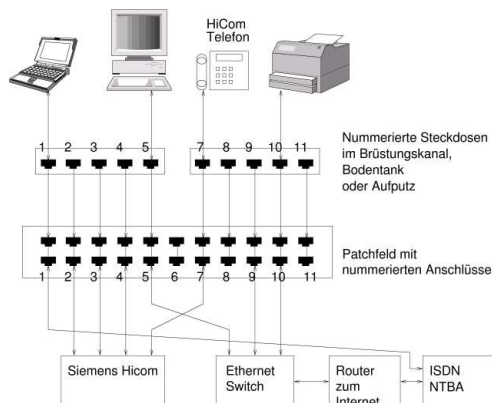
1. Netscape merkt nicht, ob eine 3COM- oder INTEL-Netzkarte oder ein Modem für die Verbindung sorgt; denn
  - der Unterschied betrifft nur den Link Layer, aber
  - Netscape arbeitet im Application Layer.
2. Protocol Stacks sind praktisch.
3. Der TCP/IP Protocol Stack eignet sich hervorragend für die Kommunikation zwischen gleichen und unterschiedlichen Architekturen.

kurze Pause...

# Kapitel 2: Link-, Network- und Transport- Layer

## Media Layer

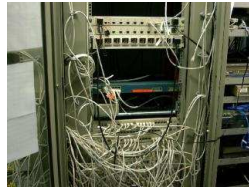
Der Media Layer stellt die physikalische Verbindung zwischen zwei Punkten mittels Kupfer, Glasfaser, Laserstrecke, Infrarotblitz, Richtfunk etc her.



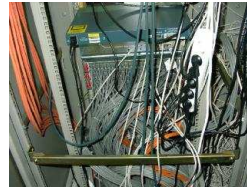
# Media Layer: Patchfeld

Das Patchfeld erleichtert Veränderungen an der Verkabelung.

Vorderansicht



Rückansicht

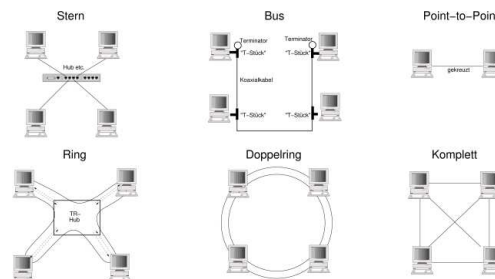


## Link Layer

Der Link Layer besteht aus zwei Sub-Layers mit folgenden Aufgaben:

- Low Level: Hardware, die Frames in Signale umsetzt und eine *Topologie* erwartet. (z.B. *Modem\**, *NIC\**). Erkennt ggf. Leitungsschäden oder Kollisionen.
- Hi Level (am Beispiel *Ethernet*):
  - alle Sendungen in Frames teilen und *CRC\** anhängen.
  - eindeutige Empfänger- und Absender-ID mitsenden (*MAC\**-Adresse)
  - Ein Protokoll zum Aufsetzen des Network Layer anbieten

## Low Level Link Layer: Topologie



## Low Level Link Layer: NIC, Modem, ISDN Adapter

Hardware, die den Computer mit dem Netz verbindet.

NIC



MODEM



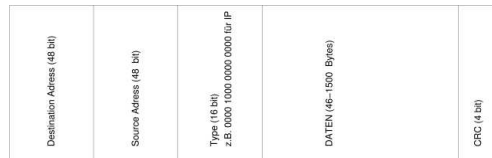
## Low Level Link Layer: Hub, Concentrator, Repeater

Der Hub leitet alle Frames an die jeweils anderen Ports weiter.  
Es gibt zwei Arten von Hubs:

- Passive Hub = Concentrator: Luxus Lüsterklemme.
- Active Hub = Repeater: verstärkt die Signale entweder analog oder digital.



## Hi Level Link Layer: Spezifikation Ethernet-Frame

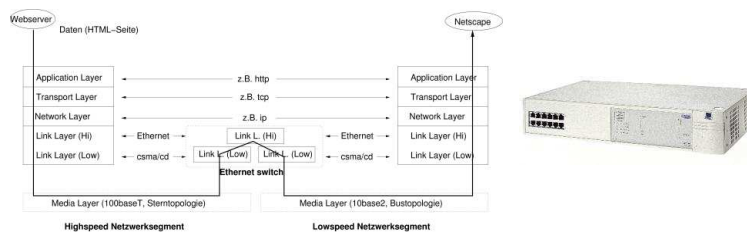


Weitere mögliche Protokolle: z.B. PPP<sub>\*</sub>, SLIP<sub>\*</sub>, IEEE<sub>\*</sub> 802.2 und 802.3, je nach verwendeter Hardware.

## Hi Level Link Layer: Switch, Bridge

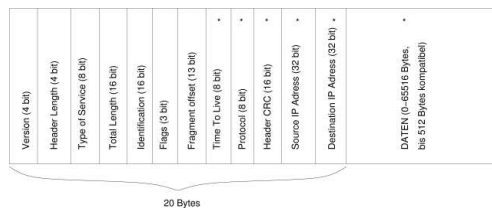
Bestimmen den Ausgangsport anhand der MAC-Adressen.

Im Gegensatz zum Switch wandelt die Bridge zusätzlich verschiedene Low Level-Protokolle ineinander um. (z.B. von 100baseT auf Token Ring).



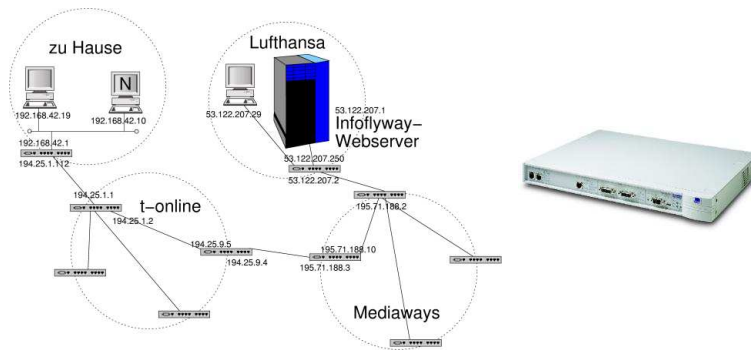
## Network Layer

Das typische Network-Layer Protokoll des TCP/IP Stack ist das IP in Version 4.



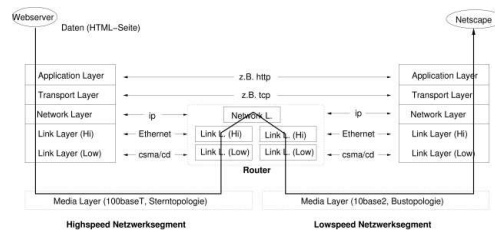
## Network Layer: Routing

Der Network Layer implementiert das Routing.



IP-Router

## Network Layer: Router



„Na und, das können Hubs oder Switches auf Link Ebene doch auch...?“

## Network Layer: Hubs und Switches sind keine Router

„Das können Hubs oder Switches auf Link Ebene doch auch...?“

Theoretisch: ja.

Aber in der Praxis: kaum möglich.

Der ausschließliche Einsatz von Hubs oder Switches hätte in einem großen Netz wie z.B. dem Internet, folgende Nachteile:

- alle Switches müssten vorher lernen, wo welche MAC-Adresse erreichbar ist.
- Hubs würden das Netz mit unnötigen Frames fluten und Kapazität verschwenden

## Network Layer: IP-Adresse

Jede Netzschnittstelle erhält

- eine eindeutige 32bit IP-Adresse (zusätzlich zur 48 bit MAC-Adresse),
- eine *Routing Tabelle*

Zur besseren Lesbarkeit notiert man die IP-Adresse als dezimale Bytes.

Bytes können Werte zwischen 0 und 255 annehmen.

Beispiel: aus 11000000101010000010101000001100 wird 192.168.42.12

Wie lautet die Dezimaldarstellung der IP-Adresse 10001101000000100000000100000001?

## Network Layer: Routing Tabelle

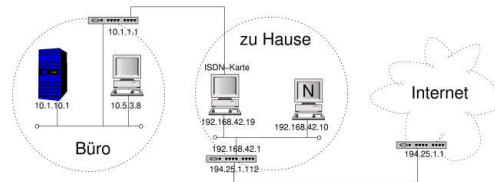
Die Routing Tabelle einer Netzschnittstelle enthält pro Zeile folgende Einträge:

- Ziel-Adresse: Eine fremde IP-Adresse.
- Gateway-Adresse: IP-Adresse eines Routers, über den man zur Zieladresse gelangt.
- *Netzmaske* (32 bits): gibt an, welche Bits eines Empfängers mit der Ziel-Adresse übereinstimmen müssen, damit ein IP Datagram über diesen Gateway geleitet wird.

Beispiel:

Ziel	Gateway	Netzmaske
192.168.42.0	0.0.0.0	255.255.255.0
10.0.0.0	192.168.42.19	255.0.0.0
0.0.0.0	192.168.42.1	0.0.0.0

## Network Layer: Routing Tabelle Beispiel



Routing Tabelle von 192.168.42.10:

Ziel	Gateway	Netzmaske
192.168.42.0	0.0.0.0	255.255.255.0
10.0.0.0	192.168.42.19	255.0.0.0
0.0.0.0	192.168.42.1	0.0.0.0

## Network Layer: Routing-Tabelle : Netzmaske

Die Bits der IP-Adresse, an deren Position die Netzmaske 1 ist, bilden die Netzadresse, der Rest den Hostpart. Beispiele:

- 255.255.255.255 für Hostrouting,
- 255.255.255.0 für Routing in Klasse C Netze
- 255.255.0.0 für Routing in Klasse B Netze
- 255.0.0.0 für Klasse A Netze
- 0.0.0.0 für default route

	192.	168.	42.	10.
IP-Adresse:	11000000	10101000	00101010	00001010
	Netzwerkadresse=192.168.42.0			Hostpart=10
	255.	255.	255.	0.
Netzmaske:	11111111	11111111	11111111	00000000
	(Klasse C)			

Wenn die Netzmaske andere Werte als 0 und 255 enthält, sagt man Sub-Netz und Subnetzmaske statt Netz und Netzmaske.

## Network Layer: Konventionen zu IP-Adressen im Internet

zur Teilnahme von IP-Adressen am Internet gibt es folgende Regeln:

- Klasse A: IP beginnt mit Bit 0 IP-Adressen von 0.0.0.0 bis 127.255.255.255
- Klasse B: IP beginnt mit Bits 10 IP-Adressen von 128.0.0.0 bis 191.255.255.255
- Klasse C: IP beginnt mit Bits 110 IP-Adressen von 192.0.0.0 bis 223.255.255.255

Zusätzlich sind die IP-Adressen 10.\*.\*.\*, 172.16.\*.\*-172.31.\*.\* und 192.168.\*.\* für private Netze reserviert und im Internet verboten. (Quelle: RFC1597\*)

Wieso?

# Network Layer: Diagnosetools ping und traceroute

```

# ping 192.168.42.1
PING 192.168.42.1 (192.168.42.1): 56 data bytes
64 bytes from 192.168.42.1: icmp_seq=0 ttl=255 time=0.5 ms
64 bytes from 192.168.42.1: icmp_seq=1 ttl=255 time=0.5 ms
64 bytes from 192.168.42.1: icmp_seq=2 ttl=255 time=0.5 ms
64 bytes from 192.168.42.1: icmp_seq=3 ttl=255 time=0.5 ms
64 bytes from 192.168.42.1: icmp_seq=4 ttl=255 time=0.5 ms
64 bytes from 192.168.42.1: icmp_seq=5 ttl=255 time=0.5 ms
64 bytes from 192.168.42.1: icmp_seq=6 ttl=255 time=0.4 ms
64 bytes from 192.168.42.1: icmp_seq=7 ttl=255 time=0.4 ms
64 bytes from 192.168.42.1: icmp_seq=8 ttl=255 time=0.4 ms

```

```

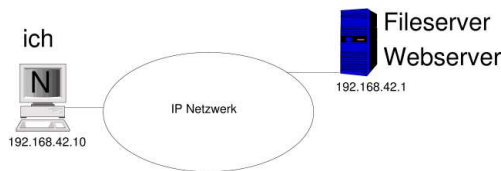
# traceroute lnk.pob.de
traceroute to lnk.pob.de (195.185.210.3), 30 hops max, 38 byte packets
 1  gate.jfranken.de (192.168.42.1)  0.583 ms  0.499 ms  0.489 ms
 2  ffm2-d2-2.atn-bb.de (62.104.212.35)  39.691 ms  24.507 ms  24.264 ms
 3  03-0-4.ffm2-gwr.atn-bb.de (62.104.212.5)  24.275 ms  24.135 ms  24.455 ms
 4  de-cix.frankfurt.nacamar.net (194.31.212.30)  24.359 ms  24.618 ms  24.441 ms
 5  atm6-0-6.hhi.nacamar.net (194.112.25.90)  41.642 ms  41.620 ms  41.642 ms
 6  fe0-0.hh0.nacamar.net (194.162.54.162)  42.016 ms  42.286 ms  41.665 ms
 7  lksynet-gw.nacamar.net (62.26.251.2)  44.307 ms  44.951 ms  44.368 ms
 8  if02.pob.de (195.185.126.178)  48.869 ms  48.890 ms  48.665 ms
 9  lnk.pob.de (195.185.210.3)  46.056 ms  45.778 ms  45.620 ms

```

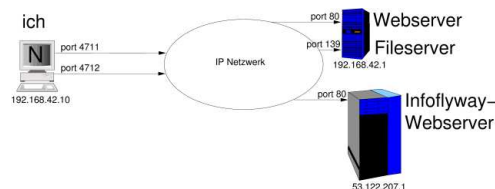
## Transport Layer

IP hat zwei Probleme:

- Die Daten sind nicht geprüft (außer ggf. der 4 bit Ethernet-Frame-CRC).
- 192.168.42.1 weiß nicht, ob er die Daten an das Webserverprogramm oder das Fileserverprogramm leiten soll:



## Transport Layer: Ports



## Transport Layer: UDP

Source Port (16 bit)	Destination Port (16 bit)	UDP Length (16 bit)	UDP Checksum (16 bit)	DATEN (0-65509)
----------------------	---------------------------	---------------------	-----------------------	-----------------

UDP Datagram

## Transport Layer: UDP Nachteile

In einigen Anwendungen (z.B. telnet, ftp) hätte UDP gravierende Nachteile, weil

- die Verbindung nach erfolgter Sendung (z.B. eines Tastendrucks) abbricht und
- keine Bestätigung des Erhalts vorgesehen ist.



Ideen zur Lösung?

## Transport Layer: TCP

TCP bietet die verlässliche, sitzungorientierte Übertragung beliebig langer Nachrichten.

Source Port (16 bit)	Destination Port (16 bit)	Sequence Number (32 bit)	Acknowledgment Number (32 bit)	Header Length (4 bit)	reserved (6 bit)	URGENT (1 bit)	ACKNOWLEDGMENT (1 bit)	PUSH (1 bit)	RESET (1 bit)	SYN (1 bit)	FINISHED (1 bit)	Window Size (16 bit)	TCP Checksum (16 bit)	Urgent Pointer (16 bit)	DAATEN (0-65508 Bytes, 1472 für Ethernet-Kompatibilität)
----------------------	---------------------------	--------------------------	--------------------------------	-----------------------	------------------	----------------	------------------------	--------------	---------------	-------------	------------------	----------------------	-----------------------	-------------------------	--

Pause...

## Kapitel 3: Application Layer

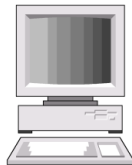
### Hostnames

Zur Vereinfachung kann man

- jeder IP-Adresse einen oder mehrere Hostnamen zuordnen (z.B. `www`)
- jedem Hostname einen Domainnamen zuordnen (z.B. `lufthansa.de`)

Die Darstellung Hostname.Domainname heißt FQDN\*.

Beispiel:



IP=192.168.42.1  
Hostname= `www`  
Domainname= `jfranken.de`  
FQDN= `www.jfranken.de`

### Hosts Resolving

Zur Erstellung eines TCP-Segments oder einer UDP-Message wird die IP-Adresse benötigt. Daher haben Anwendungen die Aufgabe, die IP-Adresse aus dem Hostname zu ermitteln. Dieser Vorgang heißt Auflösung oder resolving.

Typische Vorgehensweisen hierbei sind die *Hosts*- und die *DNS*-Methode.

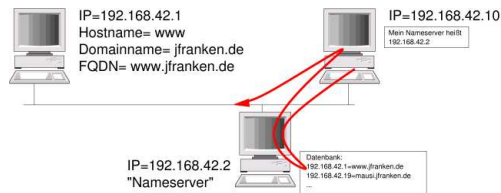
Hosts Methode:



### DNS Resolving

Der Nameserver ist ein Programm auf einem Netzrechner. Über das DNS\*-Protokoll bietet es anderen Netzteilnehmern auf Port 53

- die IP-Adresse zu einem Hostname (forward mapping)
- die Hostnamen zu einer IP-Adresse (reverse mapping, pointer queries)
- die Weiterleitung der Auflösung an andere Nameserver (forwarding)



## Well known ports

Protokoll	Port
telnet	23
daytime	13
smtp	25
http	80
ftp	21
dns	53
https	443
pop3	110
imap2	143
netbios-ns	137
netbios-ssn	139
snmp	161
printer	515
ircd	6667
oracle	1521

Die Umsetzung von Portnamen in Portnummern erledigt die `services`-Datei.

## telnet

```

X:mausi # telnet 192.168.42.10
Trying 192.168.42.10...
Connected to www.jfranken.de.
Escape character is '^Z'.
Last login: Fri Sep 22 14:19:15 2000 from mausi.jfranken.de on tty00
Linux gate 2.2.17 #12 Wed Sep 13 19:19:41 CEST 2000 1684.usdosen
su mail:
franken@gate:/home/franken # mail
Input:
Connection closed by foreign host.
  
```

Annotations:

- portname > services-Auflösung (points to 192.168.42.10)
- hostname > DNS-Auflösung (points to www.jfranken.de)
- Sitzung auf gate (points to franken@gate)

## daytime

```

X:mausi # telnet 192.168.42.10
Trying 192.168.42.10...
Connected to mausi.jfranken.de.
Escape character is '^Z'.
Fri Sep 22 14:19:21 2000
Connection closed by foreign host.
  
```

Annotations:

- portname > services-Auflösung (points to 192.168.42.10)
- hostname > DNS-Auflösung (points to mausi.jfranken.de)
- Ausgabe des daytime-Servers (points to Fri Sep 22 14:19:21 2000)

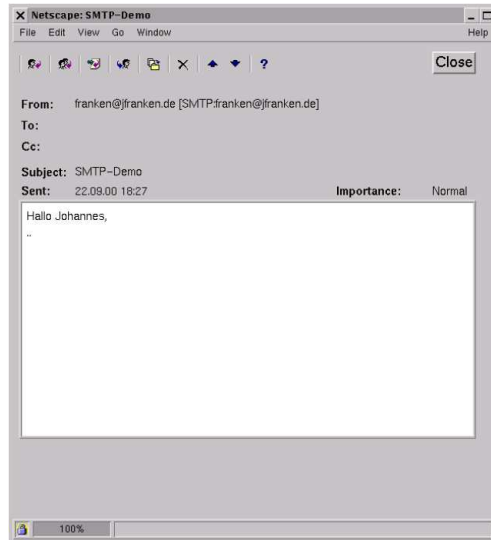
## smtp

```

SMTP-Stellung öffnen # telnet gate smtp
Trying 192.168.42.1...
Connected to gate.jfranken.de.
Escape character is '^I'.
220 gate.jfranken.de ESMTP Bxian 3.12 #1 Fri, 22 Sep 2000 18:26:10 +0200
Postfix die Absender HELO banner:
250 gate.jfranken.de Hello franken at hamster.jfranken.de [192.168.42.10]
Absender der Mail MAIL FROM <franken@franken.de>
250 <franken@franken.de> is syntactically correct
Empfänger der Mail RCPT TO: <johannes.franken@ihaynet.com>
250 <johannes.franken@ihaynet.com> is syntactically correct
Beginn des Mail-Headers DATA
354 Enter message ending with "." on a line by itself
Mail-Header Subject: SMTP-Demo
Leerzeile
Mail-Text Hallo Johannes,
...
Termin:
250 OK id=130Vex-000087-00
SMTP-Stellung beenden quit
221 gate.jfranken.de closing connection
Connection closed by foreign host.

```

# smtp



# http

```

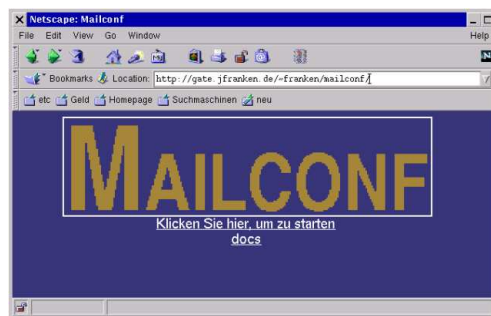
Aufbau einer Http-Stellung # telnet gate 80
Trying 192.168.42.1...
Connected to gate.jfranken.de.
Escape character is '^I'.
URL aufrufen -> Postfach-Version GET http://gate.jfranken.de/franken/mailconf/ HTTP/1.0

Http-Header
HTTP/1.1 200 OK
Date: Fri, 22 Sep 2000 17:07:54 GMT
Server: Apache/1.3.3 (Ubuntu Debian/GNU)
Last-Modified: Fri, 22 Sep 2000 17:07:07 GMT
Etag: "/httpd-1.3.3-980917"
Accept-Ranges: bytes
Content-Length: 33
Content-Type: text/html
Connection: close
Content-Disposition: inline
X-Pad: avoid browser bug

Http-Body-HTML-Date
<HTML>
<HEAD><TITLE>Mailconf</TITLE></HEAD>
<BODY bgcolor="#993399" text="#FFFFFF" link="#FFFFFF" vlink="#FFFFFF"
+>
<P>
<A href="cgi-bin/mail.pl?config_action=mailconf.pl" width="80%">
Klicken Sie hier, um zu starten
</A>
</P>
</BODY>
</HTML>
Server beendet Stellung
Connection closed by foreign host.

```

# http



# ftp

```
Aufuhf des FTP-Clients mit Server | xterm
# ftp gate
Connected to gate.jfranken.de.
220 ProFTPD 1.2.9gpld Server. (Debian) (gate.jfranken.de)
Name (gate:franken): franken
331 Password required for franken.
Password:
330 User franken logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
FTP> put tcpip.dvi
100K: tcpip.dvi remote: tcpip.dvi
200 PORT command successful.
150 Opening BINARY mode data connection for tcpip.dvi.
226 Transfer complete.
35848 bytes sent in 0.00 secs (18841.7 kb/s)
ftp> exit
221 Goodbye.
#
```

## Kapitel 4: Fachliteratur

### Fachliteratur

- die RFCs (stets aktuell auf <http://www.rfc-editor.org/>)
- Andrew S. Tanenbaum, *Computer Networks*, Prentice Hall
- W. Richard Stevens, *Unix Network Programming*, Prentice Hall
- W. Richard Stevens, *TCP/IP Illustrated, Volume 1 und 3*, Addison-Wesley
- Paul Albitz & Cricket Liu, *DNS and BIND*, O'Reilly
- Johannes Franken, *DNS & BIND GE-PACKT*, MITP
- Anatol Badach, *Datenkommunikation mit ISDN*, International Thomson Publishing
- Oxford University Press, *Computer Lexikon*, Sybex