Netzprogrammierung
Internet-Dienste

Prof. Dr.-Ing. Robert Tolksdorf
Freie Universität Berlin
Institut für Informatik
Netzbasierte Informationssysteme
mailto: tolk@inf.fu-berlin.de
http://www.robert-tolksdorf.de

Überblick

- Internet Dienste
- Mail
- FTP

Was ist das Internet

- Eine weltweiter *Verbund von Rechnern*, die über Netze Daten austauschen können.
  - Hardware-bezogene Sicht
  - Zusammenschalten von lokalen Netzen zum Internet
  - Dabei notwendige Verarbeitung von Datenpaketen
- Eine *Protokollfamilie*
  - Netzbezogene Sicht
  - Protokollspezifikationen
- Ein *offenes System*, in dem Dienste genutzt und angeboten werden können.
  - Nutzungs- und anwendungsbezogen
  - Beschreibt die Anwendungsmöglichkeiten des Internet

Internet als Protokollfamilie

- *Request For Comments*-Dokumente (RFC) definieren alle technischen Aspekte des Internet
- RFC 1738 :
  T. Berners-Lee, L. Masinter, und M. McCahill. Uniform Resource Locators (URL). RFC 1738, Internet Engineering Task Force, December 1994.
- Internet Engineering Taskforce IETF erstellt RFCs
  http://www.ietf.org/rfc.html
- Standardisierungsprozess ist als RFC standardisiert:
  The Tao of IETF: A Novice's Guide to the Internet Engineering Task Force, RFC 3160, August 2001

IETF Arbeitsfelder (7/02)

- Applications Area
- General Area
- Internet Area
- Operations and Management Area
- Routing Area
- Security Area
- Sub-IP Area
- Transport Area

IETF Workinggroups
Applications Area (06/06)

Freie Universität Berlin

atompub	Atom Publishing Format and Protocol
calsify	Calendaring and Scheduling Standards Simplification
crisp	Cross Registry Information Service Protocol
eai	Email Address Internationalization
imapext	Internet Message Access Protocol Extension
ldapbis	LDAP (v3) Revision
lemonade	Enhancements to Internet email to support diverse service environments
ltru	Language Tag Registry Update
opes	Open Pluggable Edge Services
sieve	Sieve Mail Filtering Language
usefor	Usenet Article Standard Update
webdav	WWW Distributed Authoring and Versioning
widex	Widget Description Exchange Service

Ausgewählte Standards und RFCs

Freie Universität Berlin

Protokoll	Beschreibung	RFCs	STD
	Internet Official Protocol Standards	1880	1
	Assigned Numbers	1700	2
	Host Requirements - Communications	1122	3
	Host Requirements - Applications	1123	3
IP	Internet Protocol	791	5
IP	Subnet Extension	950	5
IP	Broadcast Datagrams	919	5
IP	Broadcast Datagrams with Subnets	922	5
ICMP	Internet Control Message Protocol	792	5
IGMP	Internet Group Multicast Protocol	1112	5
UDP	User Datagram Protocol	768	6
TCP	Transmission Control Protocol	793	7
TELNET	Telnet Protocol	854, 855	8
FTP	File Transfer Protocol	959	9

Ausgewählte Standards und RFCs

Freie Universität Berlin

Protokoll	Beschreibung	RFCs	STD
SMTP	Simple Mail Transfer Protocol	821	10
SMTP-SIZE	SMTP Service Ext for Message Size	1870	10
SMTP-EXT	SMTP Service Extensions	1869	10
MAIL	Format of Electronic Mail Messages	822	11
CONTENT	Content Type Header Field	1049	11
NTPV2	Network Time Protocol, Version 2	1119	12
DOMAIN	Domain Name System	1034, 1035	13
DNS-MX	Mail Routing and the Domain System	974	14
SNMP	Simple Network Management Protocol	1157	15
SMI	Structure of Management Information	1155	16
Concise-MIB	Concise MIB Definitions	1212	16
MIB-II	Management Information Base-II	1213	17
NETBIOS	NetBIOS Service Protocols	1001, 1002	19
ECHO	Echo Protocol	862	20

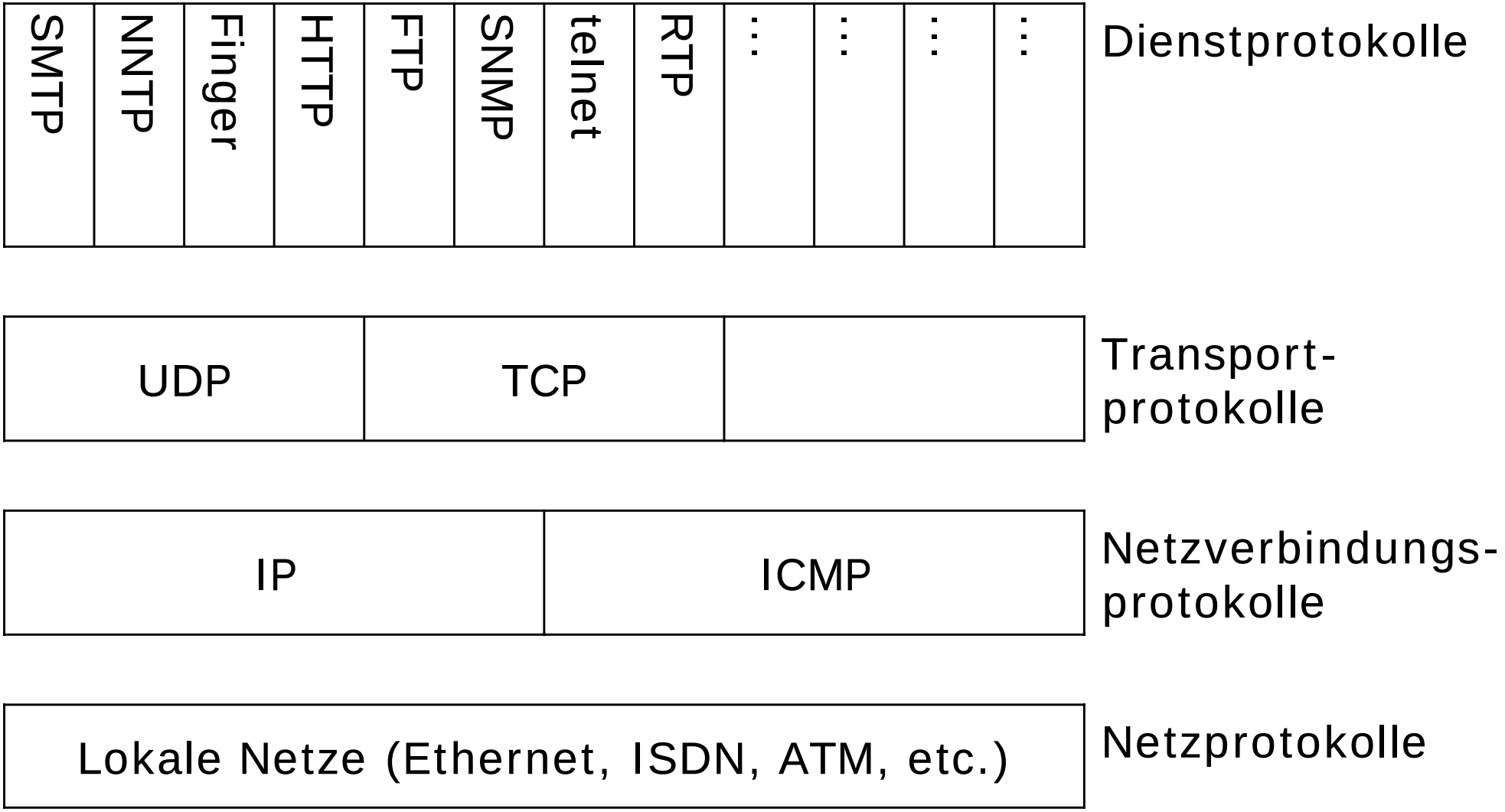
Ausgewählte Standards und RFCs

Freie Universität Berlin

Protokoll	Beschreibung	RFCs	STD
DISCARD	Discard Protocol	863	21
CHARGEN	Character Generator Protocol	864	22
QUOTE	Quote of the Day Protocol	865	23
USERS	Active Users Protocol	866	24
DAYTIME	Daytime Protocol	867	25
TIME	Time Server Protocol	868	26
TFTP	Trivial File Transfer Protocol	1350	33
RIP	Routing Information Protocol	1058	34
TP-TCP	ISO Transport Service on top of the TCP	1006	35
ETHER-MIB	Ethernet MIB	1643	50
PPP	Point-to-Point Protocol (PPP)	1661	51
PPP-HDLC	PPP in HDLC Framing	1662	51
IP-SMDSIP	Datagrams over the SMDS Service	1209	52

Internet-Protokolle und -Dienste

- Einordnung von Internet-Protokollen:



Internet als dienstorientiertes offenes System

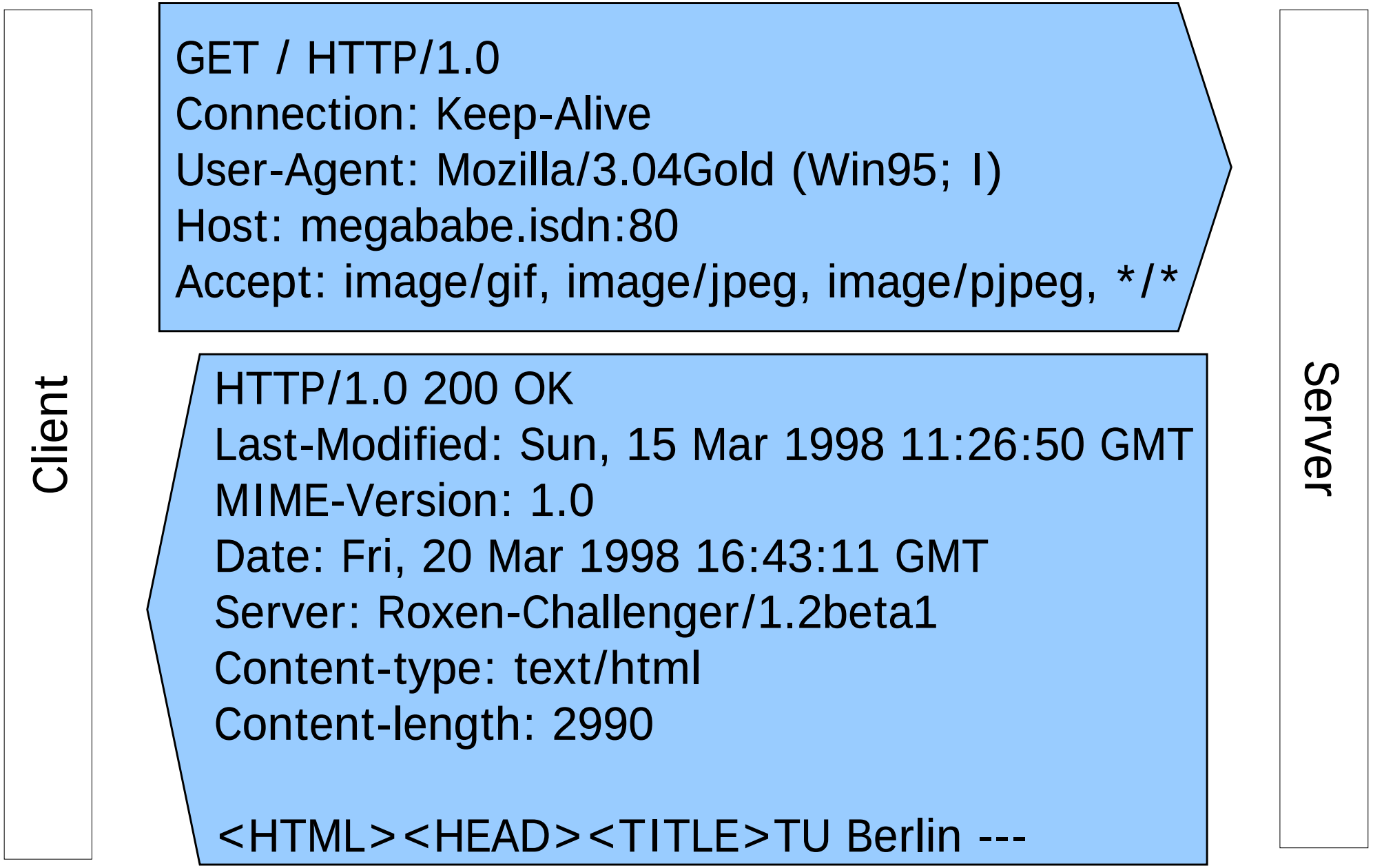
Freie Universität Berlin

- Internet Dienste sind (zumeist) definiert durch
  - Aufgabe
  - Portnummer auf dem der Dienst angeboten wird
  - Transportprotokoll (TCP oder/und UDP)
  - Protokoll

- Z.B.: Web Dienst
  - Übertragen von HTML Seiten
  - Port 80
  - TCP
  - HTTP

- Z.B.: Usenet Dienst
  - Übertragen von News
  - Port 119
  - TCP
  - NNTP

Beispiel: HTTP Protokoll



Mail

Simple Mail Transfer Protocol SMTP

Freie Universität Berlin

- Aufgabe:
  Transfer von Mails zwischen Mail-Client beim Absender und Mailserver beim Empfänger
- Ports:
  25 zur kompletten Protokollabwicklung
- Transportprotokoll:
  TCP
- Protokoll:
  J. Klensin, Editor. *Simple Mail Transfer Protocol*, April 2001. RFC 2821, http://www.ietf.org/rfc/rfc2821.txt

Beispiel von Hand

- Zum Erproben von Protokollen kann man auch per Hand das Protokoll mit dem Server „sprechen“
- Notwendig: Socket-Verbindung
- telnet Programm verbindet Standardein- und -ausgabe mit Socket
- >telnet *Rechnername Portnummer*
- Daytime abfragen:

```
athos:/home/datsche/tolk [30]% telnet np.ag-nbi.de 13
Trying 85.10.200.21...
Connected to np.ag-nbi.de.
Escape character is '^]'.
22 NOV 2005 10:06:09 CET
Connection to np.ag-nbi.de closed by foreign host.
```

Beispiel von Hand

Freie Universität Berlin

```
athos:/home/datsche/tolk [28]% telnet mail 25
Trying 160.45.40.10...
Connected to leibniz.
Escape character is '^]'.
220 math.fu-berlin.de ESMTP
HELO
250 math.fu-berlin.de
MAIL from: mailfaker@inf.fu-berlin.de
250 ok
RCPT To: tolk@inf.fu-berlin.de
250 ok
DATA
354 go ahead
From: The Mail Faker

Hallo - das ist gar nicht von mir...
.
250 ok 1132650117 qp 20018
quit
221 math.fu-berlin.de
Connection to leibniz closed by foreign host.
```

Beispiel von Hand

Freie Universität Berlin

```
From mailfaker@inf.fu-berlin.de Tue Nov 22 09:02:00 2005
Return-Path: <mailfaker@inf.fu-berlin.de>
Delivered-To: tolk@inf.fu-berlin.de
Received: (qmail 20318 invoked from network); 22 Nov 2005 10:02:00 +0100
Received: from lusin.mi.fu-berlin.de (HELO mi.fu-berlin.de) (160.45.113.91)
  by leibniz.math.fu-berlin.de with SMTP; 22 Nov 2005 10:02:00 +0100
Received: (qmail 3403 invoked by uid 9804); 22 Nov 2005 10:02:00 +0100
Received: from localhost (HELO mi.fu-berlin.de) (127.0.0.1)
  by localhost with SMTP; 22 Nov 2005 10:01:58 +0100
Received: (qmail 3389 invoked by uid 9804); 22 Nov 2005 10:01:58 +0100
Received: from leibniz.math.fu-berlin.de (HELO math.fu-berlin.de) (160.45.40.10)
  by lusin.mi.fu-berlin.de with SMTP; 22 Nov 2005 10:01:58 +0100
Received: (Qmail 20018 invoked from network); 22 Nov 2005 10:01:35 +0100
Received: From athos.mi.fu-berlin.de (HELO ) (160.45.110.55)
  by leibniz.math.fu-berlin.de with SMTP; 22 Nov 2005 09:01:35 -0000
From: The Mail Faker
X-Envelope-Sender: mailfaker@inf.fu-berlin.de
X-Virus-Scanned: by AMaViS 0.3.12pre7-L32[3392](NAI-uvscan@mi.fu-berlin.de)
X-Remote-IP: 127.0.0.1

Hallo - das ist gar nicht von mir...
```

Beispiel per Programm

Freie Universität Berlin

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

class SMTPMail {

    SMTPMail(String host, String from, String to, String content)
    throws java.io.IOException {

    String message;
    // zu diesem Rechner verbinden
    Socket socket = new Socket(host,25);
    // Ströme vorbereiten
    BufferedReader in =
        new BufferedReader(new InputStreamReader(
                            socket.getInputStream()));
    DataOutputStream out = new
          DataOutputStream(socket.getOutputStream());
```

Beispiel per Programm

Freie Universität Berlin

```
    // Mail schicken
    out.writeBytes("HELO\r\n"+
                   "MAIL from: "+from+"\r\n"+
                   "RCPT To: "+to+"\r\n"+
                   "DATA\r\n"+
                   "From: "+from+"\r\n\r\n"+
                   content+
                   "\r\n.\r\n"+
                   "QUIT");
    // Antwort lesen und ausgeben
    socket.setSoTimeout(500);
    try {
       while ((message = in.readLine())!=null)  System.out.println("Got "+message);
    } catch (SocketTimeoutException ste) {}
    // alles schliessen und Schluss.
    out.close();
    in.close();
    socket.close();
   }
   public static void main (String args[]) throws java.io.IOException {
    new SMTPMail("mail.inf.fu-berlin.de", args[0], args[1], args[2]);
   }
}
```

Beispiel per Programm

```
>java SMTPMail mailfaker@inf.fu-berlin.de tolk@inf.fu-
  berlin.de "Hallo - ich bin es nicht"
Got 220 math.fu-berlin.de ESMTP
Got 250 math.fu-berlin.de
Got 250 ok
Got 250 ok
Got 354 go ahead
Got 250 ok 1132651343 qp 2507
```

- Nicht nachmachen!

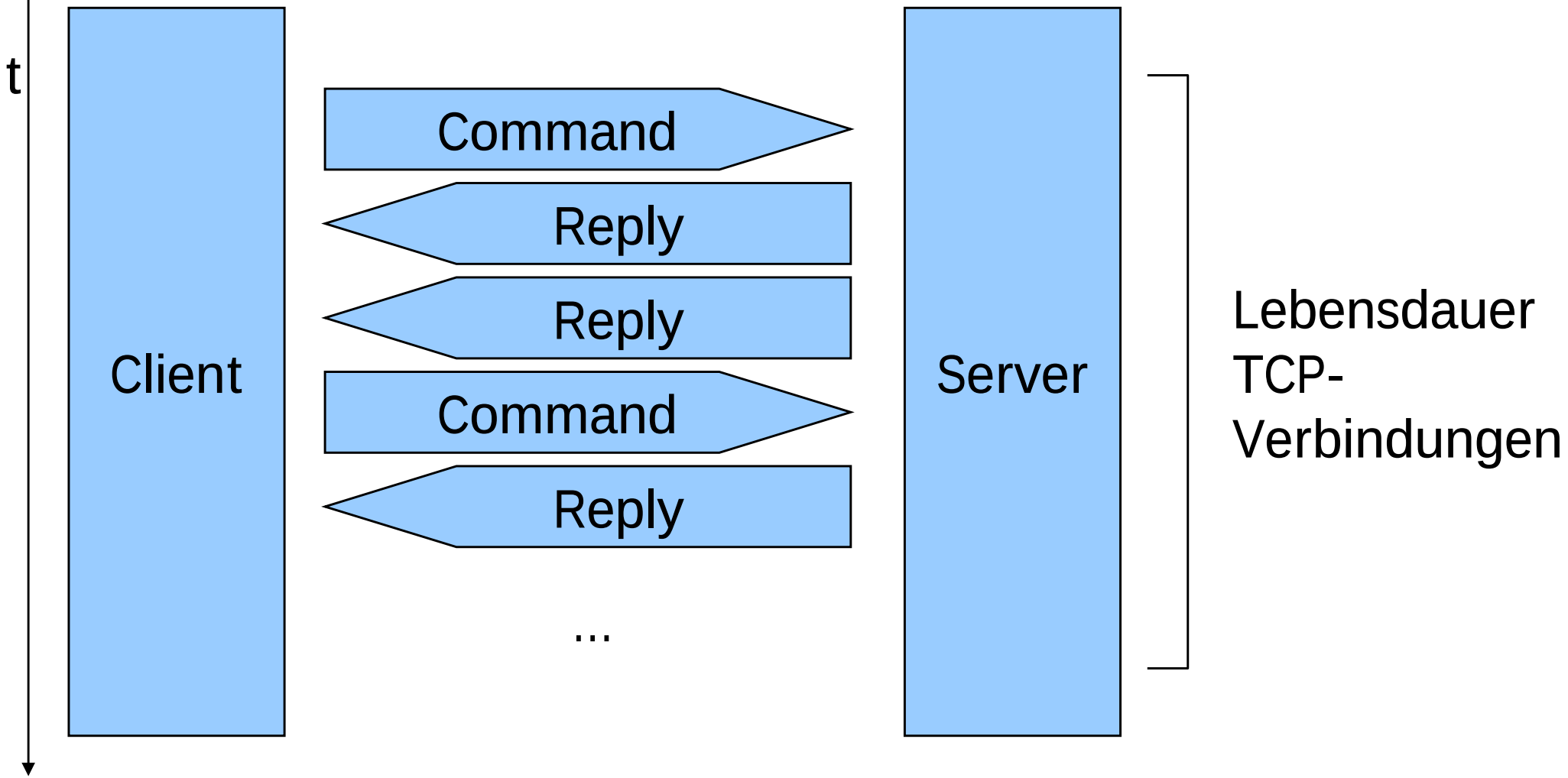
FTP

File Transfer Protocol

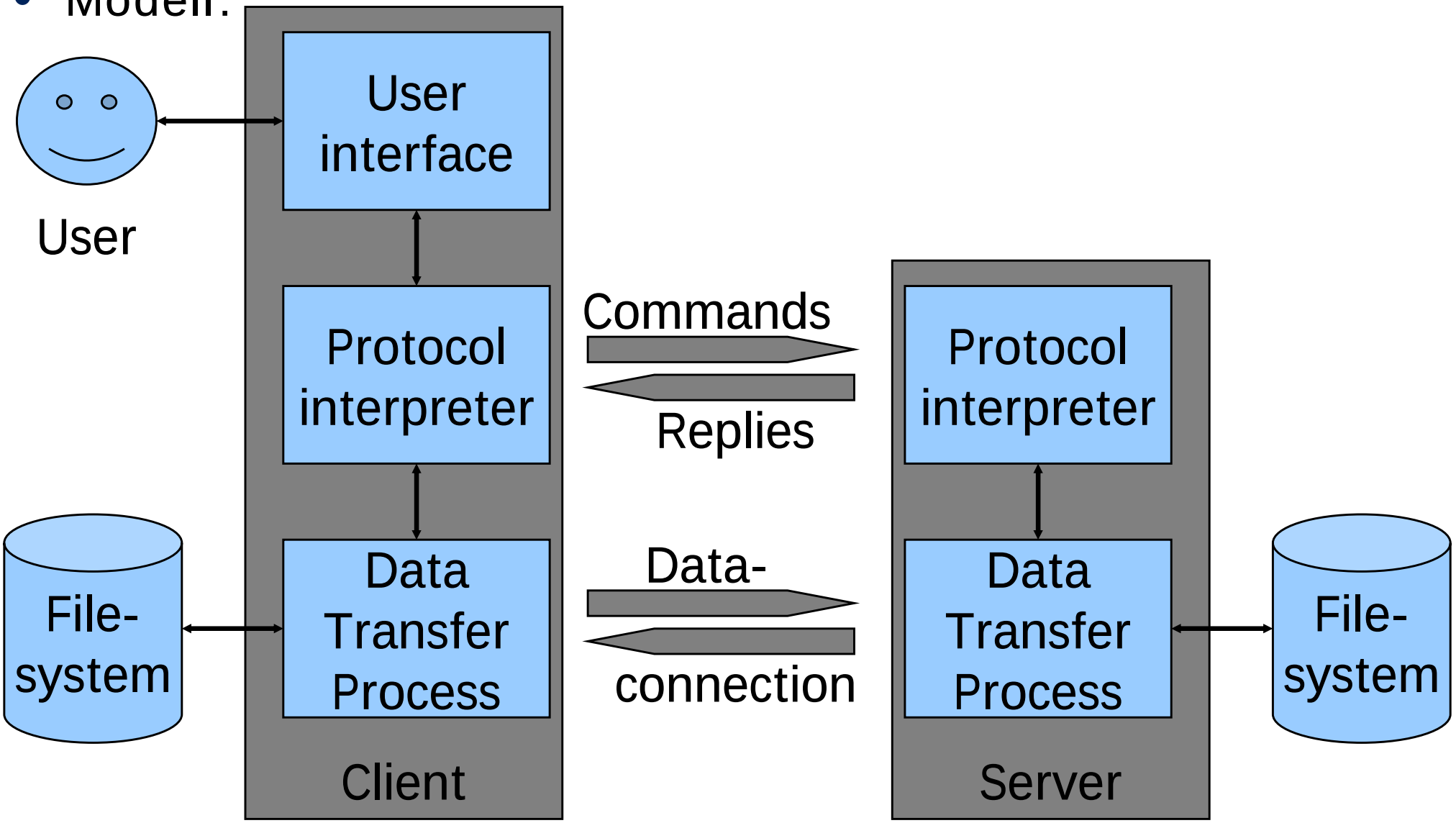
- Aufgabe:
  Transfer von Dateien zwischen FTP-Servern und Clients
- Ports:
  21 ist für FTP Kontrollverbindung reserviert
  Weitere Ports sind für FTP Datenverbindung reserviert
- Transportprotokoll:
  TCP
- Protokoll:
  J. Postel und J. Reynolds. *FILE TRANSFER PROTOCOL (FTP)*, Oktober 1985. RFC 959, http://www.ietf.org/rfc/rfc959.txt

FTP

- Zustandshaltiges Protokoll
- Request mit Response beantwortet



- Modell:


User
User interface
Protocol interpreter
Data Transfer Process
Client
File-system
Commands
Replies
Data-connection
Protocol interpreter
Data Transfer Process
Server
File-system

Beispielsitzung per Hand

Freie Universität Berlin

```
Connected to caramba.
220 ftp.cs.tu-berlin.de FTP server ready.
Name (ftp:tolk): ---> USER tolk
331 Password required for tolk.
---> PASS **********************
230 User tolk logged in.
ftp> ---> PORT 130,149,17,167,185,53
200 PORT command successful.
---> LIST
150 Opening ASCII mode data connection for /bin/ls.
total 33264
drwxr-xr-x  52 tolk     flp          8704          Jul 22 17:20 .
drwxr-sr-x  44 root     root         2560          Jun 25 14:23 ..
-rw-r--r--      1 tolk     flp          164352        Jul 16 09:22 NBI.ppt
[...]
226 Transfer complete.
12491 bytes received in 0.13 seconds (97.37 Kbytes/s)
```

Login/Passwort-Eingabe

Eingabe "dir"

Beispielsitzung per Hand

Freie Universität Berlin

```
ftp> ---> PORT 130,149,17,167,185,54
200 PORT command successful.
---> RETR test
150 Opening ASCII mode data connection for test (6 bytes).
226 Transfer complete.
local: test remote: test
6 bytes received in 0.052 seconds (0.11 Kbytes/s)
ftp> ---> PORT 130,149,17,167,185,55
200 PORT command successful.
---> RETR nofile
550 nofile: No such file or directory.
ftp> ---> QUIT
221-You have transferred 6 bytes in 1 files.
221-Total traffic for this session was 13058 bytes in 2 transfers.
221-Thank you for using the FTP service on ftp.cs.tu-berlin.de.
221 Goodbye.
```

Eingabe "get test"

Eingabe "get nofile"

Eingabe "quit"

Beispiel per Programm

Freie Universität Berlin

- Einfacher FTP-Client
- Aufruf:
  java FtpClient ftp.inf.fu-berlin.de tolk@inf.fu-berlin.de pub readme
- (Dieser Klient hier ist nicht in gutem OO-Stil geschrieben!)

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

public class FtpClient {

    static Socket command;
    static BufferedReader br;
    static PrintWriter pw;
```

Beispiel per Programm

```
public static String getResponse(String expected) {
 String response, code, text;
 try {
    response=br.readLine();
    System.out.println(">"+response);
    // Antwortcode extrahieren
    code=response.substring(0,3);
    // bei YZX- weiterlesen bis XYZ  (Leerzeichen!)
    if (response.charAt(3)=='-') {
        text=br.readLine();
        System.out.println(">"+text);
        while ((text.length()<3) || !text.substring(0,4).equals(code+" ")) {
           text=br.readLine();
           System.out.println(">"+text);
        }
    }
    // Eigentlich: Antwortcode verarbeiten....
    return response;
 } catch (Exception e) {
    System.out.println(e); return "";}
}
```

Beispiel per Programm

- Eine Socketverbindung verwendet einen Puffer auf dem sendenden Rechner
- Wenn der Puffer hinreichend voll ist, wird ein Datenpaket auf den Weg gebracht
- Beim Schreiben in Java-Ströme wird zuerst der Puffer gefüllt
- Damit Gegenseite Zeichenkette erhält muss der Puffer „ausgespült“ werden -> flush()-Aufruf notwendig

```
public static void sendCommand(String command) {
   System.out.println("<"+command);
   // Kommando senden und rausschicken
   pw.println(command);
   pw.flush();
}
```

Beispiel per Programm

```
public static void main(String[] argv) {
 Socket data;
 byte[] buffer = new byte[1024];
 int read;
 try {
    command = new Socket(argv[0], 21);
    pw = new PrintWriter(command.getOutputStream());
    br = new BufferedReader(new InputStreamReader(command.getInputStream()));
    // Begrüssung
    getResponse("220");
    // Einloggen
    sendCommand("USER anonymous");
    // Passwortanforderung
    getResponse("331");
    sendCommand("PASS " + argv[1]);
    // Ok, eingeloggt
    getResponse("230");
    // Verzeichnis wechseln
    sendCommand("CWD " + argv[2]);
    // Ok, gewechselt
    getResponse("250");
    // Binäre Übertragung wählen
    sendCommand("TYPE I");
    // Ok
    getResponse("200");
```

Beispiel per Programm

- FTP Datenkommunikation
  - Aktiv:
    - Server nimmt Verbindung zu Klient auf
    - Klient teilt Port vorher mit
  - Passiv
    - Klient kontaktiert Server
    - Server teilt Port vorher mit
- Problem bei aktivem Modus:
  - Klient muss an Port lauschen
  - Schwierigkeit: Firewall
  - Schwierigkeit: Private Netze mit gemeinsamer IP-Nummer
- Unser Klient wählt passiven Modus

Beispiel per Programm


```
        // Client baut Datenverbindung auf, Server ist passiv
        sendCommand("PASV");
        // 227 Entering Passive Mode (160,45,117,6,152,81).
        String[] field=(getResponse("227").substring(27)).split("[,)]");
        // Datei anfordern
        sendCommand("RETR " + argv[3]);
        // Datensocket öffnen
        data= new Socket(field[0]+"."+field[1]+"."+field[2]+"."+field[3],
                         Integer.parseInt(field[4])*256+Integer.parseInt(field[5]));
        // Daten lesen
        while ((read=data.getInputStream().read(buffer))!=-1) {
            System.out.println(new String(buffer,0,read));
        }
        // Abschlussmeldung holen
        getResponse("");
        // Verabschieden
        sendCommand("QUIT");
    } catch (IOException iOExc) {
        System.err.println(iOExc.getMessage());
    }
  }
}
```

Beispiel per Programm

```
java FtpClient ftp.inf.fu-berlin.de tolk@inf.fu-berlin.de pub readme
>220-
> ---------------------------------------------
> login as anonymous or ftp , no user login allowed
> ---------------------------------------------
>
>220 ProFTPD 1.2.10 Server (ftp.mi.fu-berlin.de) [160.45.117.6]
<USER anonymous
>331 Anonymous login ok, send your complete email address as your password.
<PASS tolk@inf.fu-berlin.de
>230-
>
> Welcome, archive user anonymous@fock.inf.fu-berlin.de !
>
> ---------------------------------------------
> You are connected to the anonymous ftp server
> ftp.mi.fu-berlin.de (160.45.117.6) at
> Freie Universität Berlin (Germany/Europe).
[...]
> If you do have problems, please try using a dash (-) as the first
> character of your password -- this will turn off the continuation
> messages that may be confusing your FTP client.
>230 Anonymous access granted, restrictions apply.
```

Beispiel per Programm


```
<CWD pub
>250 CWD command successful
<TYPE I
>200 Type set to I
<PASV
>227 Entering Passive Mode (160,45,117,6,152,115).
<RETR readme
This is only a local File-Hierarchy of 'math.fu-berlin.de' !
                                    = = = =
For public-domain Software and other unspecific
Information use server 'ftp.fu-berlin.de'
[...]
 - UNIX ...(use the central server)

 ftp://ftp.fu-berlin.de/pub/unix/security/openssh/


Thanks   <stucki@math.fu-berlin.de>

>150 Opening BINARY mode data connection for readme (588 bytes)
<QUIT
```

Kommandos zur Zugriffskontrolle

- Einloggen:
  - USER NAME (USER)
  - PASSWORD (PASS)
  - ACCOUNT (ACCT)
- Navigation
  - CHANGE WORKING DIRECTORY (CWD)
  - CHANGE TO PARENT DIRECTORY (CDUP)
  - STRUCTURE MOUNT (SMNT)
- Sitzungsmanagent
  - REINITIALIZE (REIN)
  - LOGOUT (QUIT)

Kommandos zu Übertragungsparametern

- Datenverbindung
  - DATA PORT (PORT)
    Datenport, falls nicht Standard
  - PASSIVE (PASV)
    Server wartet auf Datenverbindung
- Formate
  - REPRESENTATION TYPE (TYPE)
    Übertragungsrepräsentation (ASCII, Wortlänge etc.)
  - FILE STRUCTURE (STRU)
    Dateistruktur (File, Records, Seiten)
  - TRANSFER MODE (MODE)
    Übertragungsmodus (Strom, Blöcke, Komprimierung)

Kommandos zur Übertragung

Freie Universität Berlin

- Dateitransfers
  - RETRIEVE (RETR)
  - STORE (STOR)
  - STORE UNIQUE (STOU)
  - APPEND (with create) (APPE)
  - ALLOCATE (ALLO) Platz reservieren
  - RENAME FROM (RNFR) + RENAME TO (RNTO)
  - ABORT (ABOR)
  - DELETE (DELE)
- Verzeichnisse
  - REMOVE DIRECTORY (RMD)
  - MAKE DIRECTORY (MKD)
  - PRINT WORKING DIRECTORY (PWD)
  - LIST (LIST)
  - NAME LIST (NLST)

Kommandos zur Übertragung

- Informationen
  - SYSTEM (SYST)
  - SITE PARAMETERS (SITE)
  - STATUS (STAT)
  - HELP (HELP)
- NOOP (NOOP)

Antwort Codes der Form xyz

- Antwortart-Codes für x:

1yz	Vorläufig positive Antwort, Server meldet sich wieder
2yz	Erfolgreiche Ausführung
3yz	Vorläufig positive Antwort, Client muß sich melden
4yz	Vorübergehen negative Antwort – erneut versuchen
5yz	Erfolglose Ausführung

- Betreff-Codes für y:

x0z	Probleme mit Anfragesyntax
x1z	Antworten auf Informationsanfragen
x2z	Antworten bzgl. Verbindungen
x3z	Antworten bzgl. Authentifizierungen
x5z	Antworten bzgl. Status des Dateisystems

Syntax

- 200 Command okay.
- 500 Syntax error, command unrecognized.
- 501 Syntax error in parameters or arguments.
- 202 Command not implemented, superfluous at this site.
- 502 Command not implemented.
- 503 Bad sequence of commands.
- 504 Command not implemented for that parameter.

Information

- 110 Restart marker reply.
- 211 System status, or system help reply.
- 212 Directory status.
- 213 File status.
- 214 Help message.
- 215 NAME system type
  (Where NAME is an official system name from the list in the Assigned Numbers document)

Verbindungen

- 120 Service ready in nnn minutes.
- 220 Service ready for new user.
- 221 Service closing control connection.
- 421 Service not available, closing control connection.
- 125 Data connection already open; transfer starting.
- 225 Data connection open; no transfer in progress.
- 425 Can't open data connection.
- 226 Closing data connection.
- 426 Connection closed; transfer aborted.
- 227 Entering Passive Mode (h1,h2,h3,h4,p1,p2).

Authentifizierung

- 230 User logged in, proceed.
- 530 Not logged in.
- 331 User name okay, need password.
- 332 Need account for login.
- 532 Need account for storing files.

Dateisystem

- 150 File status okay; about to open data connection.
- 250 Requested file action okay, completed.
- 257 "PATHNAME" created.
- 350 Requested file action pending further information.
- 450 Requested file action not taken. File unavailable
- 550 Requested action not taken. File unavailable
- 451 Requested action aborted. Local error in processing.
- 551 Requested action aborted. Page type unknown.
- 452 Requested action not taken. Insufficient storage space
- 552 Requested file action aborted. Exceeded storage allocation
- 553 Requested action not taken. File name not allowed.

Zusammenfassung

Zusammenfassung

1. Internet als Protokollfamilie
   1. RFCs der IETF definieren Internetdienste
   2. Dienste durch Aufgabe, Port, TP, Protokoll definiert
2. Mail
   1. Kommunikation über Kommandos zum Server
   2. Header frei wählbar
3. FTP
   1. Kommunikation über zwei Sockets
   2. Protokoll