Rechnernetze und verteilte Systeme

Internet-Dienste

Kapitel 10

Anwendungsschicht im Prinzip

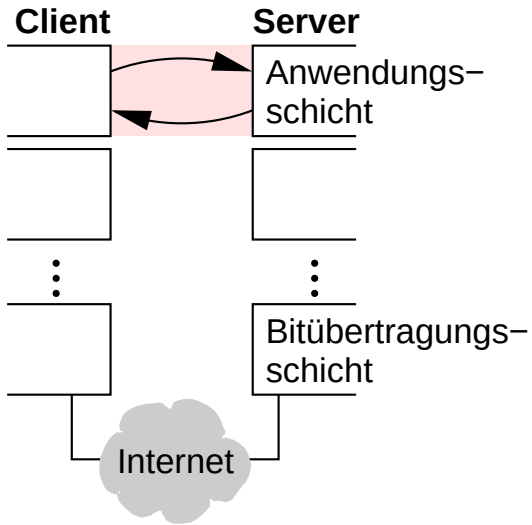
- Anwendungen...
  - werden auf vernetzten Rechnern ausgeführt
  - tauschen Nachrichten aus, um einen Dienst zu erbringen
- Protokolle der Anwendungsschicht...
  - sind Teil der Anwendungen
  - nutzen darunterliegende Protokolle
  - spezifizieren die Art der ausgetauschten Nachrichten
  - spezifizieren die Aktionen, die auf Nachrichten folgen

- Client-Server Modell bei vielen vernetzten Anwendungen

Client kontaktiert Server und fordert Dienst an, z.B. Auslieferung eines HTML-Dokuments (HTTP)

Server beantwortet Anfrage (d.h. überträgt HTML-Dokument)

Protokoll spezifiziert u.a. Syntax der Anfrage/Antwort



Rechnernetze und verteilte Systeme
Internet-Dienste

Eigenschaften und Anforderungen

Kapitel 10.1

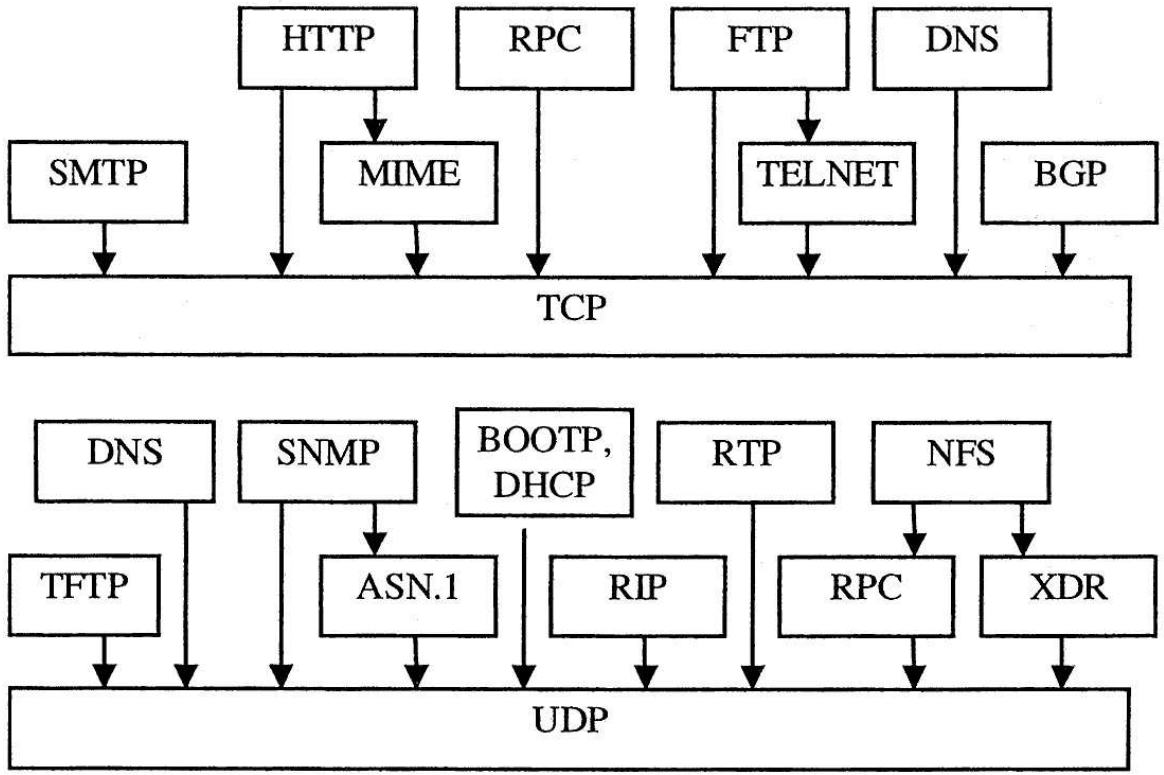
- Ziele der Anforderungen
  - Unbeschwerte Nutzung eines gegebenen Dienstes
  - Dimensionierung des Netzes bzw. der Übertragungskanäle
  - Wahl von Transportverfahren
- Toleranz bezüglich Datenverlust
  - Audio-, Telephonie-, Video-Anwendungen, Spiele in gewissem Rahmen verlussttolerant
  - viele gängige Anwendungen erfordern verlustfreie, korrekte Übertragung (WWW, FTP, Email. . .)
- Benötigte Übertragungsrate
  - minimale Rate gefordert bei Multimedia-Anwendungen (z.B. Audio-/Video-Ströme; Zusammenhang mit Puffergrößen in Clientapplikation)
  - andere Anwendung haben „elastische” Anforderungen; sie passen sich der verfügbaren Übertragungsrate an (z.B. Datei gegebener Größe wird scheller/langsamer übertragen)
- Verzögerung
  - maximale Verzögerung gefordert bei Echtzeitanwendungen (z.B. Telephonie, Spiele)

10.1 Eigenschaften und Anforderungen

Typische Anforderungen einiger Anwendungen

Anwendung	empfindlich bezüglich		
	Verlust	Übertragungsrate	Verzögerung
File Transfer	ja	elastisch	nein
E-Mail	ja	elastisch	nein
WWW	ja	elastisch (wenige Kbps)	nein
Audio/Video (Echtzeit)	tolerant	Audio: wenige Kbps – 1Mbps. Video: 10 Kbps bis 5Mbps	Ja, einige hundert Millisek.
Audio/Video (gespeichert)	tolerant	wie oben	Ja, wenige Sekunden
Interaktive Spiele	tolerant	wenige Kbps bis 10 KBps	Ja, einige hundert Millisek.
Finanz-anwendungen	ja	elastisch	ja und nein

- Auswahl von Transportdienst nach Dienstanforderungen
- TCP: zuverlässige Übertragung; höhere Kosten wegen Verbindunsaufbau, Quittungen etc
- UDP: nicht zuverlässig aber „sparsamer” (kleinerer Header, verbindungslos)



Wichtige funktionale Anforderung aller Dienste:
Abbildung von Namen auf Adressen

- DNS (Domain Name System)
  - RFCs 1034, 1035
  - dient der Abbildung von Endsystemen auf IP-Adresse
- X.500 Directory
  - Konzept der ITU-T für ein verteiltes Directory, einschließlich Verschlüsselung und Zertifizierung
- LDAP (Lightweight Directory Access Protocol)
  - RFC 1959, 2251
  - ist ein Zugriffsprotokoll für Directories gemäß X.500
  - ist weniger aufwending als OSI X.500-DAP
  - LDAP wird ergänzt durch LIPS (Lightweight Internet Person Schema) und LDIF (Lightweight Directory Interchange Format)
  - häufiger Einsatzbereich: Nutzerauthentifizierung

Rechnernetze und verteilte Systeme
Internet-Dienste

Domain Name System

Kapitel 10.2

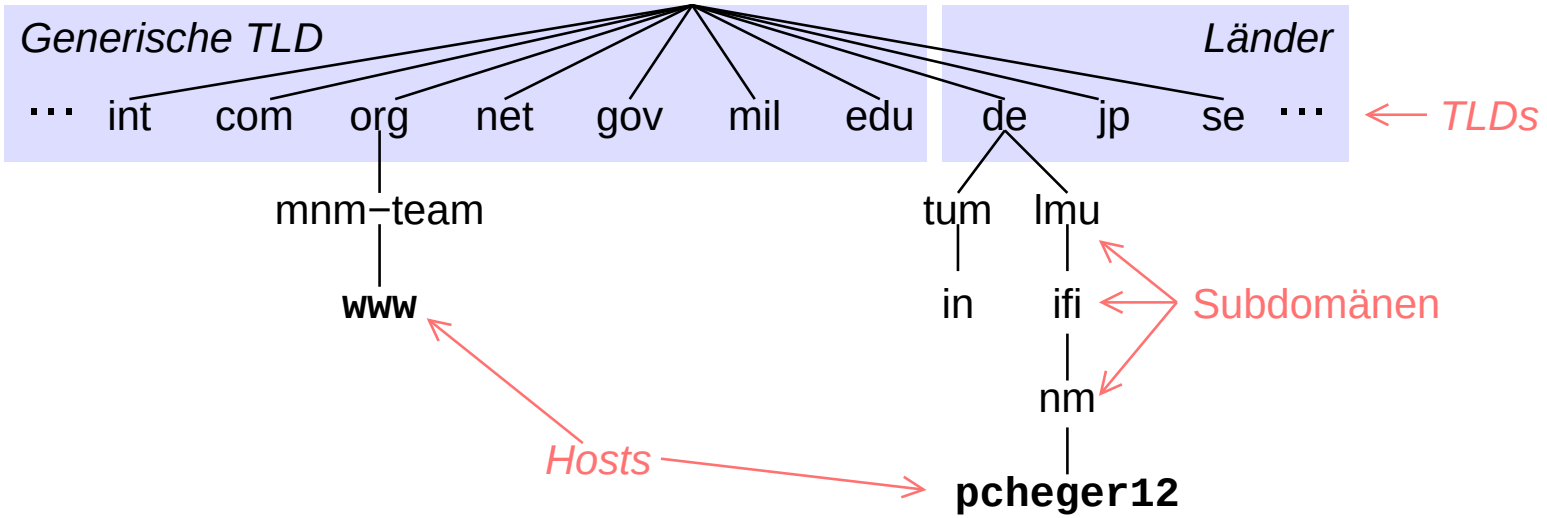
DNS

verteilte, in einer Hierarchie von Name Servern implementierte Datenbank und Protokoll der Anwendungsschicht zwischen Hosts und Name Servern

- Dienstmerkmale:
  - Abbildung von Host-Namen auf IP-Adressen
  - Host Aliasing (mnemonisch - kanonisch)
  - Mail Server Aliasing
  - Lastverteilung zw. replizierten Servern
- Funktionen eines DNS
  - Beantworten von Client-Anfragen
  - Austausch mit anderen DNS-Servern
- DNS ist kritisch für das Funktionieren des Internet
- Probleme: Bottleneck, Verkehrsvolumen, Entfernung, Pflege
- Hierarchie von DNS-Servern (wegen Skalierung) :
  - Lokaler Name Server, autoritativer Name Server, Root Name Server, vermittelnder Name Server

10.2 Domain Name System

Namensraum

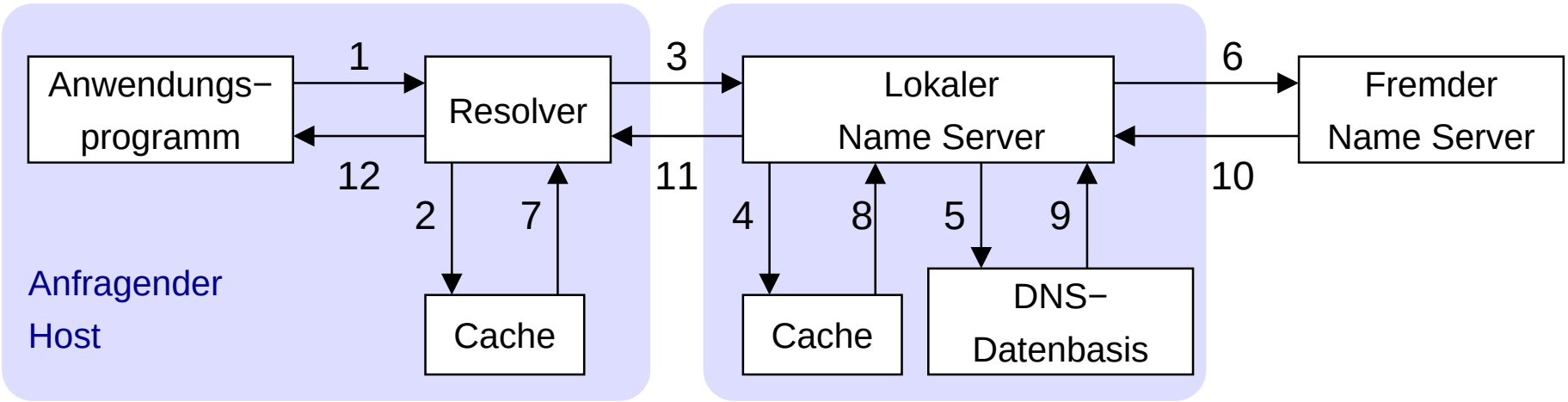


- Hierarchisch, Baumstruktur
- Top Level Domains (TLD): $<$ 300 TLD festgelegt von ICANN
  - generisch, nach Zweck (z.B. `.com` →kommerzielles Unternehmen)
  - TLD für Länder (machen die meisten TLDs aus)
- Subdomains: Unterteilung in benannte Teildomänen
  - Second-level domains (z.B. `lmu.de`) von *Registraren* zugeteilt
  - Unterteilung in Subdomains kann wiederholt werden
- Host-Namen: Blätter des Baumes
- Fully Qualified Domain Name (FQDN)
  - vollständiger Name bestehend aus Hostname und Domänennamen
  - in Richtung Baumwurzel zu lesen; endet mit Punkt

- Root Name Server
  - Nur wenige (ca. 1 Dutzend) Root Name Server weltweit, gut geschützt
  - löst Domänennamen auf, liefert Adresse eines autoritativen NS
- Authoritative Name Server: löst Host-Namen einer Domäne auf
  - „authoritative”: Antwort des Servers wird als richtig angenommen
  - für jede Zone: primärer DNS-Server, sekundäre Server (Ausfallsicherheit, Lastausgleich)
- Lokaler Name Server: Namensauflösung für Nutzer eines Betreibers
  - Host muss konfiguriert werden, lokalen NS zu kennen (mit IP-Adresse!)
  - Adresse des lokalen NS wird dem Host manuell oder mit DHCP mitgeteilt
- Resolver: Softwarekomponente zur Host-lokalen Namensauflösung
- DNS Software: meist *BIND* (Berkeley Internet Name Domain)

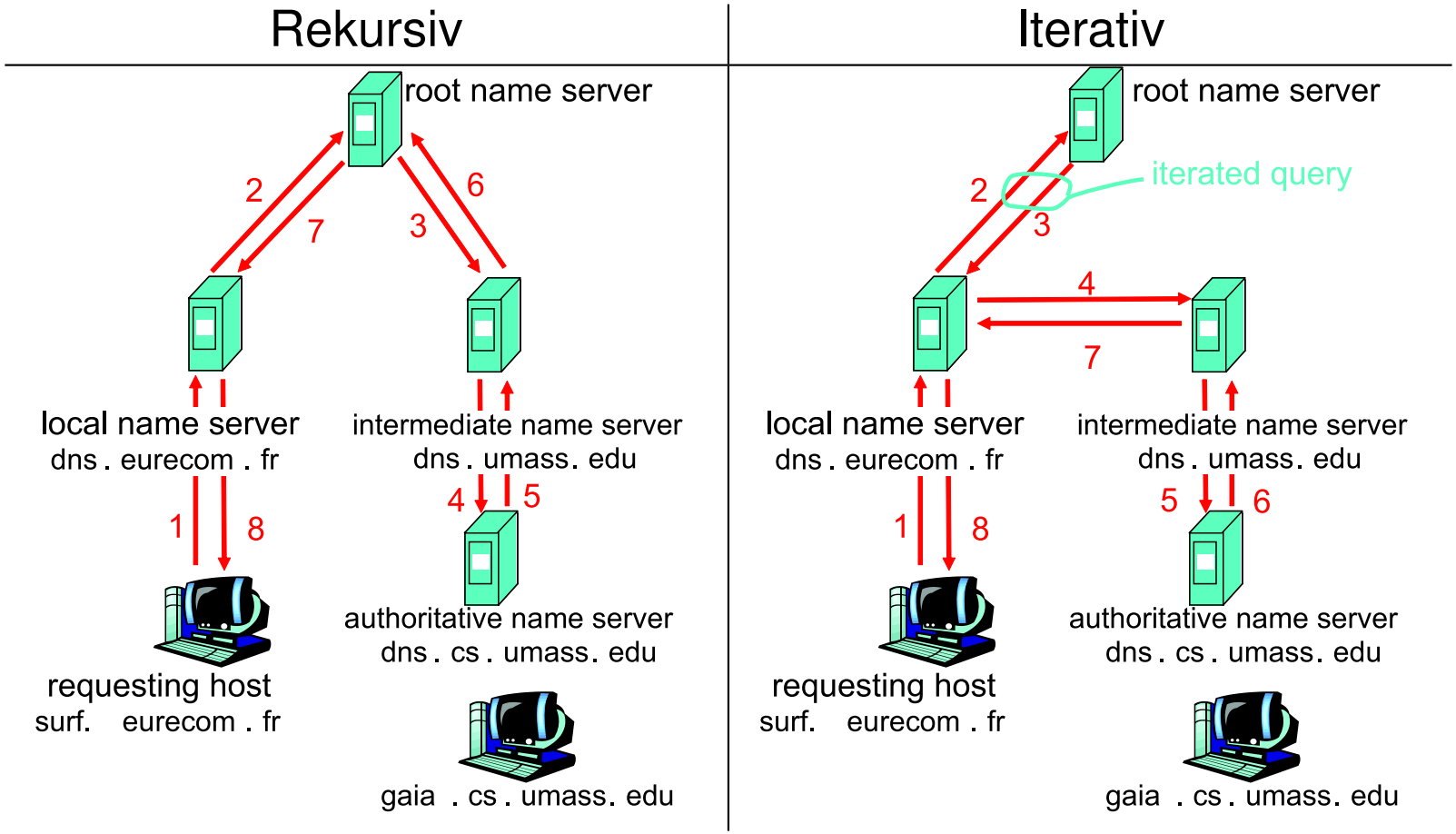
10.2 Domain Name System

Namensauflösung



- Schritte
  1. Anfrage an Resolver (lokal auf Host)
  2. Nachschlagen im Cache. Erfolg →7, 12
  3. Anfrage bei lokalem DNS-Server.
  4. Nachschlagen im Cache. Erfolg →8, 12. Update Resolver-Cache.
  5. Nachschlagen in Datenbasis. Erfolg →9, 11, 12.
  6. Anfrage an fremden DNS-Server. Antwort über 10, 11, 12.
- Inhalt der Antwort
  - gesuchte IP-Adresse, oder
  - Referenz auf DNS-Server, der den Namen auflösen kann, oder
  - „gibt es nicht”
- Antwort führt zur Auffrischung der Cache-Information

- Rekursiv: Name Server reichen Anfrage durch (→Last)
- Iterativ: Anfragen der Reihe nach



Resource Record (RR)

- Datensatz über einen Namen. Name Server liefert ihn als Anwort.
- 5-Tupel: Name, TTL, Class, Typ, Wert
- Typen (Auswahl)

A	“Address”: Name = Host, Wert = IP-Adresse (wichtigster Typ)
AAAA	: wie ’A’ für IPv6-Einträge
NS	“Name Server”: Name = Domain, Wert = Hostname eines autor. Servers
CNAME	“Canonical Name”: Wert = kanonischer Name für Alias Hostname
MX	“Mail Exchange”: Wert = Hostname eines Mailservers mit Aliasnamen

- TTL: Gültigkeitsdauer in Sekunden (Maß für Stabilität des RR)
- Class: Netztyp “IN” im Internet (selten anderer Wert. . .)

Zones

- nicht überlappende Bereiche des Namensraumes
- Zuständigkeitsbereiche für Name Server

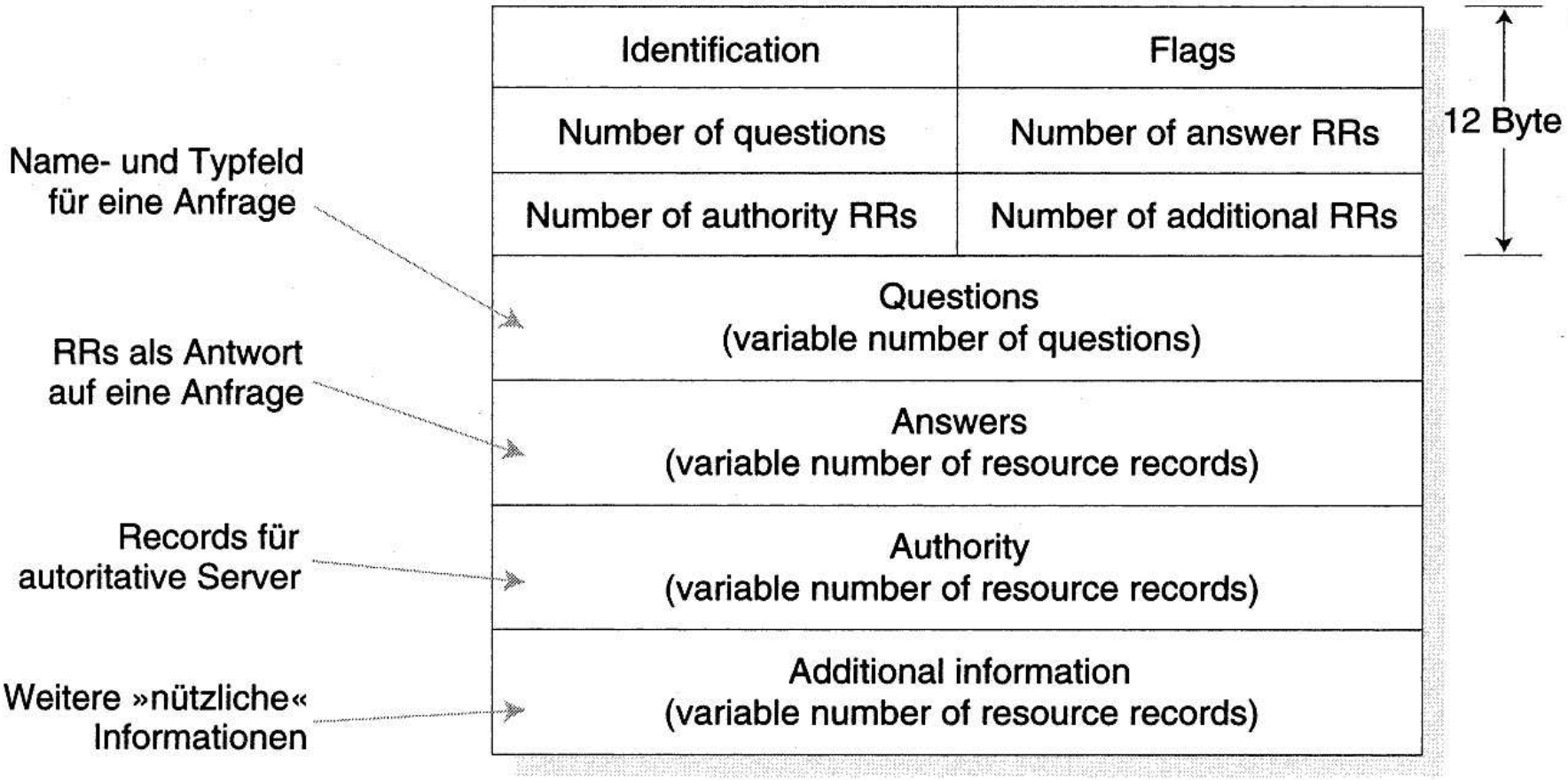
```
danciu@pcheger09:~> host www.in.tum.de
www.in.tum.de is an alias for www.informatik.tu−muenchen.de.
www.informatik.tu−muenchen.de is an alias for infoport.informatik.tu−muenchen.de.
infoport.informatik.tu−muenchen.de has address 131.159.74.65
infoport.informatik.tu−muenchen.de mail is handled by 25 mailin.informatik.tu−muenchen.de.
```

```
; <<>> DiG 9.3.4 <<>> www.in.tum.de
;; global options:  printcmd
;; Got answer:
;; −>>HEADER<<− opcode: QUERY, status: NOERROR, id: 7702
;; flags: qr rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 5, ADDITIONAL: 7
;; QUESTION SECTION:
;www.in.tum.de.                 IN      A
;; ANSWER SECTION:
www.in.tum.de.          86400   IN      CNAME   www.informatik.tu−muenchen.de.
www.informatik.tu−muenchen.de. 86400 IN CNAME   infoport.informatik.tu−muenchen.de.
infoport.informatik.tu−muenchen.de.     86400   IN   A   131.159.74.65
```

[...]

Name | TTL | Class | Type | Value

- Identification: Anfrage ID
- Flags: Query/Reply, Autoritative Bit, Recursion Desired, Recursion available
- Number of: Längenangaben
- Felder (Questions, Answers . . .) enthalten Resource Records

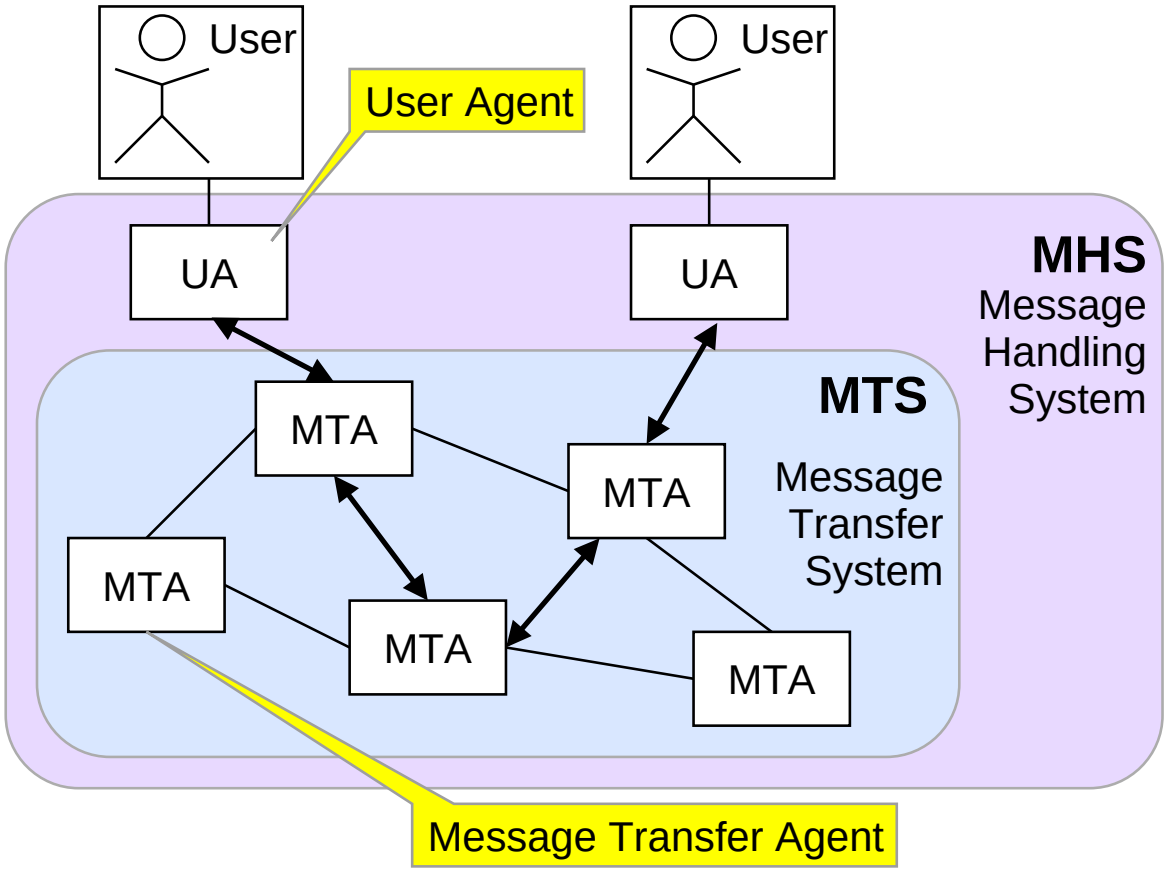


- Zonentransfer
  - Oft mehrere Name Server pro administrative Domäne: Primary NS, Secondary NS
  - Primary/Secondary NS bearbeiten Anfragen gleichgestellt (liefern autor. Antworten)
  - Konsistenzerhalt in der Datenbank der Name Server
    - Wartung einer zentralen Datei mit Resource Records
    - Unterscheidung alter/neuer Versionen mittels Seriennummer
    - Verteilung geänderter RRs mittels Zonentransfer
    - Secondary NS (Client) erfragt periodisch Änderungen beim Primary NS (Master)
- Transportprotokoll
  - Anfragen: UDP-basiert
    - Anfragen zustandslos
    - Wiederholung möglich, falls Frage oder Antwort verloren gehen
    - Verzicht auf Verbindungsaufbau →Performanz
  - Zonentransfer: TCP-basiert
    - Zuverlässigkeit von Bedeutung; größeres Datenvolumen als eine Anfrage
    - findet selten statt (im Vergleich zu Anfragen)

Rechnernetze und verteilte Systeme
Internet-Dienste

Electronic Mail

Kapitel 10.3


User
User
User Agent
UA
UA
MHS
Message
Handling
System
MTS
Message
Transfer
System
MTA
MTA
MTA
MTA
MTA
Message Transfer Agent

Begriffe aus der Internet-Welt

SMTP: Protokoll für den Transfer von E-Mail zwischen Mail-Servern, nutzt TCP und IP-Adressen.

Mail Server (Post Office): Server, der SMTP nutzt.

POP (Post Office Protocol): Protokoll zur Kommunikation zwischen Mail Server und Mail Client.

IMAP (Internet Message Access Protocol) : Nachfolger von POP.

Begriffe aus der OSI- bzw. ITU-T-Welt

X.400: Norm der ITU-T für E-Mail.

MOTIS (Message-Oriented Text Interchange System): ISO-Standard (ISO 10021) für E-Mail. Entspricht X.400.

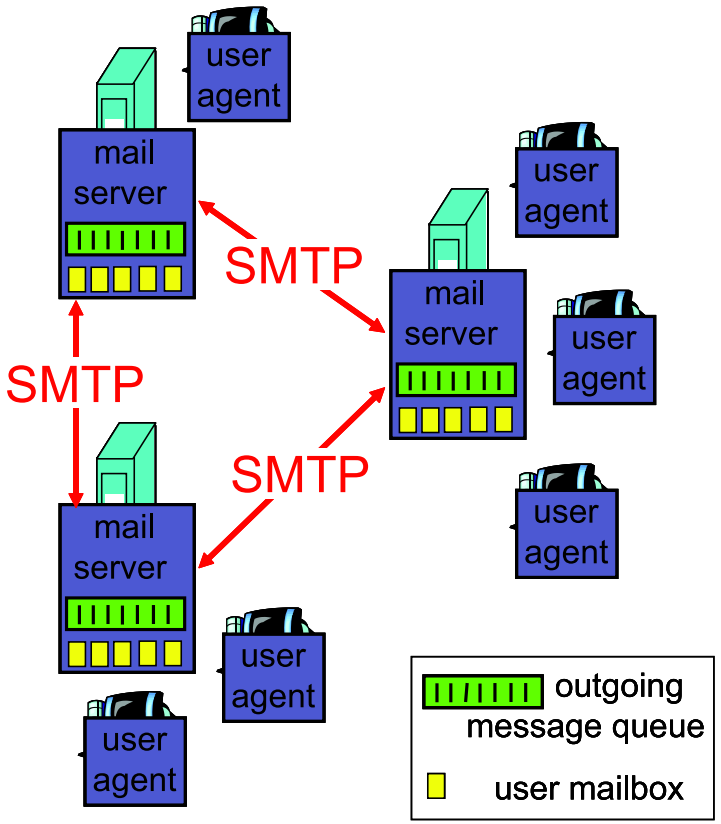
MHS (Message Handling System): bezeichnet das Gesamtsystem aus MTS, MTA, UA.

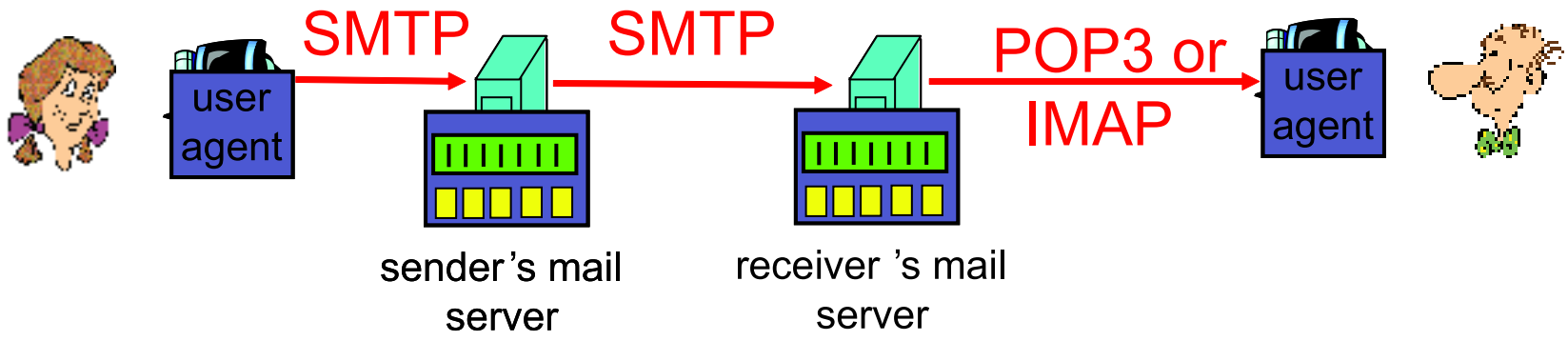
MTS (Message Transfer System): die Menge aller MTAs.

MTA (Message Transfer Agent): entspricht dem Mail Server.

UA (User Agent): entspricht dem Mail Client.

- User Agent (auch: E-Mail-Client, Mail Reader . . .)
  - z.B. Thunderbird, pine, mailx (1), Outlook . . .
  - Nutzerschnittstelle; Funktionen zum Schreiben/Verschicken und Empfangen von Nachrichten
- Mail Server
  - Aufbewahrung eingehender Nachrichten
  - Warteschlange für ausgehende Nachrichten





- Senden von Emailnachrichten: SMTP (Push-Protokoll)
  - in RFC 822 ausschließlich ASCII-Text
  - in RFC 2045/6 Erweiterung auf MIME (Binärdaten)
- Abholung von Emailnachrichten
  - POP3 (Post Office Protocol)
    - Authentifizierung/Authorisierung (User Agent, Server)
    - Übertragung der Nachrichten zum User Agent
  - IMAP (Internet Mail Access Protocol)
    - komplexer; Verwaltung mehrerer Ordner
    - Emailnachrichten können auf dem Server verwaltet werden
  - HTTP (Hypertext Transfer Protocol) für browserbasierte Emaildienste

- Direkte, TCP-basierte (Port 25) Übertragung zwischen sendendem UA/MTA und empfangendem MTA
- spezifiziert in RFC 821
- Phasen: handshaking (greeting), transfer, closure
- Command/Response-Dialog (ASCII-basiert).
- Responses: Statuscode und Phrase.

Relevanz der DNS-Informationen für Email

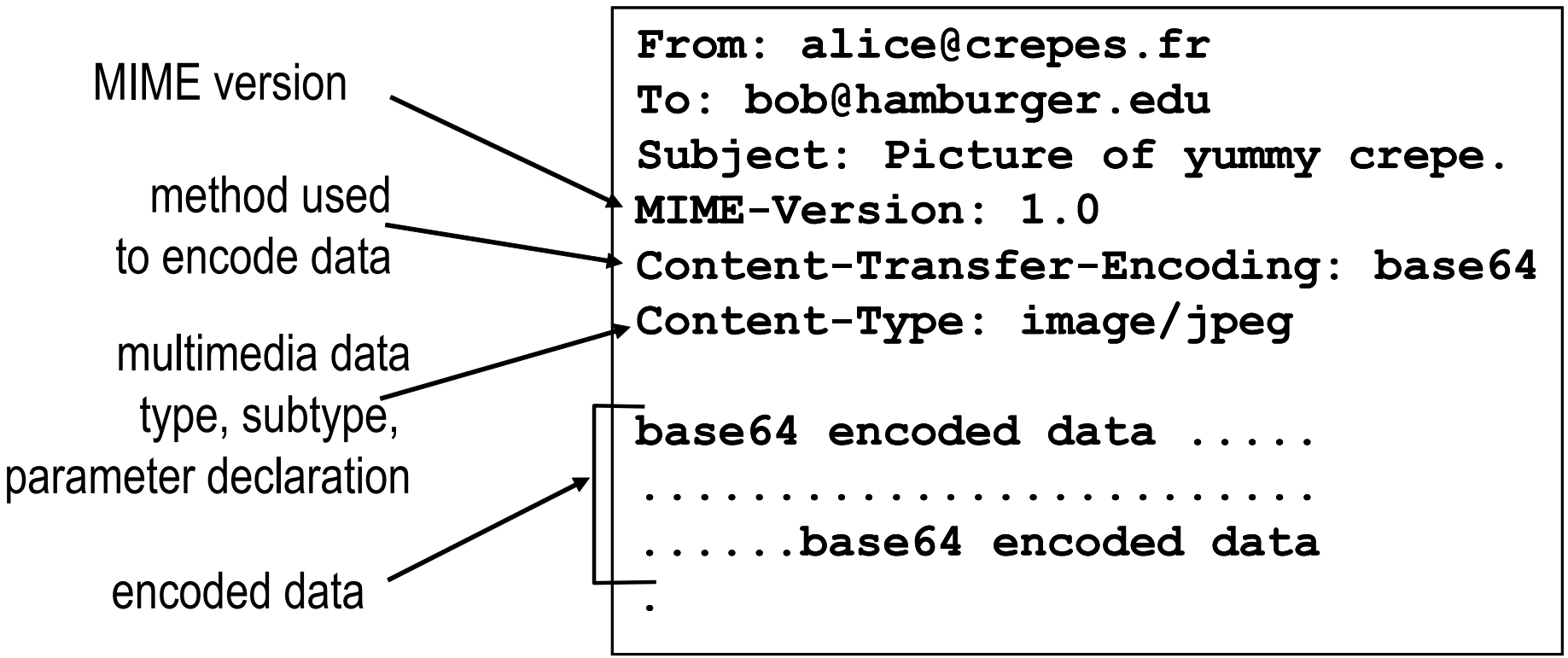
- Problem: an welchen Host soll eine an `nutzer@domain` adressierte Email übertragen werden?
- Resource Record des Typs **MX** nennt den für eine Domäne zuständigen Mailserver →Emailadressen nicht notwendig an Namensraum für Rechner gebunden

→ Adressen wie `vorname.nachname@unternehmen.com` möglich

danciu@pcheger09: > **telnet mail 25** ←Server heißt „mail”, Port ist 25
[. . .]
S: 220 mail.nm.ifi.lmu.de ESMTP Sendmail 8.12/Linux MNM 0.1; Mon, 28 Jan 2008 10:23:13 +0100 Unterstrichen: Statuscode
HELO nm.ifi.lmu.de Unterstrichen: Command
S: 250 mail.nm.ifi.lmu.de Hello pcheger09.in.nm.ifi.lmu.de, pleased to meet you
MAIL FROM:<danciu@nm.ifi.lmu.de>
S: 250 2.1.0 <danciu@nm.ifi.lmu.de>. . . Sender ok
RCPT TO:<rnp@nm.ifi.lmu.de>
S: 250 2.1.5 <rnp@nm.ifi.lmu.de>. . . Recipient ok
DATA
S: 354 Enter mail, end with “.” on a line by itself
SMTP Probelauf. ←Text der Email
. ←Einsamer Punkt
S: 250 2.0.0 m0SJNDno027963 Message accepted for delivery
QUIT
S: 221 2.0.0 mail.nm.ifi.lmu.de closing connection
Connection closed by foreign host. ←Nachricht von telnet-Client

S:. . . :SMTP-Server **fett**: Benutzereingabe rot: Kommentare

- Standard für Textnachrichten, spezifiziert in RFC 822
- Nachricht besteht aus Kopf/Header, Body sowie Abschlusszeile
- Header-Zeilen (siehe auch Beispiel in Kapitel 3)
  - To, From, Subject
  - CC, BCC ([Blind] Carbon Copy)
  - Reply-To: Emailadresse, die für Antwort benutzt werden sollte
  - Message-Id: identifiziert eine Emailnachricht in späterer Kommunikation
  - In-Reply-To, References: Verweise auf Message-Ids von Emailnachrichten
  - Received: wird von jedem vermittelnden MTA dem Header hinzugefügt
- Body
  - Nutzdaten („Brief”); nur ASCII-Zeichen zulässig
  - Letzte Zeile markiert Nachrichtenende; sie enthält nur einen Punkt ‘.’
- Erweiterung zum Transport von Multimedia-Daten
  - MIME: multimedia mail extension, RFC 2045, 2056
  - zusätzliche Information im Header →Angabe des MIME content type


MIME version
method used
to encode data
multimedia data
type, subtype,
parameter declaration
encoded data
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg
base64 encoded data .....
.....
.....base64 encoded data
.

Content Type	Bedeutung
Application	Nicht näher spezifizierte binäre Datei, Daten für ein Programm. **Subtypen:** Octet Stream (Bytefolge) und Postscript.
Audio	Sprache, Musik, Geräusche. **Subtyp:** Basic.
Image	Festbild oder Grafik. **Subtypen:** GIF, JPEG.
Message	Eine vollständige E-Mail-Nachricht oder eine Referenz auf die Nachricht (Angabe einer Datei auf einem FTP-Server). **Subtypen:** RFC 822 (nach RFC 822 codiert), Partial (Nachricht wurde für die Übertragung aufgeteilt) und External-body (Nachricht auf Server abgelegt).
Multipart	Mehrteilige Nachricht, jeder Teil hat sein eigenes Content Type und Content Transfer Encoding. **Subtypen:** Mixed: unabhängige Teile mit jeweils eigenem Type und Encoding. Alternative: Dieselbe Nachricht, in verschiedenen Repräsentationen. Parallel: Teile müssen gleichzeitig dargestellt werden, z. B. zur Synchronisation von Bild und Sprache. Digest: Jeder Teil ist eine vollständige Nachricht nach RFC 822.
Text	Unformatierter oder formatierter Text. **Subtypen:** Plain, Richtext.
Video	Bewegtbild. **Subtyp:** MPEG.

Authorisation phase

- client commands
  - user: declare username
  - pass: password
- server responses *+OK*, *-ERR*

S: +OK POP3 server ready
C: user alice
S: +OK
C: pass hungry
S: +OK user successfully logged on

Transaction phase

- list: list message numbers
- retr: retrieve message by number
- dele: delete
- quit

C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: quit
S: +OK POP3 server signing off

- Problem: Mit POP3 können Nachrichten nur abgeholt, aber nicht auf dem Server verwaltet werden.

→ Schlechte Unterstützung für nomadische Nutzer

- IMAP: Verwaltung von Emailnachrichten auf dem Server
  - Hierarchie von Ordnern (*folders*), die Nachrichten enthalten (wie Dateisystem)
  - Abrufen, Verschieben (zwischen Ordnern), Löschen von Nachrichten durch UA
  - Selektive Übertragung von Teilen von Emailnachrichten (z.B. nur Header)
- Last / Performanz
  - Verwaltungsoperationen auf Server ausgeführt → belastet Servermaschine
  - Selektive Übertragung: Abruf nur relevanter Nachrichtenteile, z.B. über Verbindung mit niedriger Bandbreite

Rechnernetze und verteilte Systeme
Internet-Dienste

World Wide Web

Kapitel 10.4

- WWW besteht aus Mengen von...
  - Client: WWW-Browser
  - Server: WWW-Server
  - Objekte: Hypertext- und Hypermedia-Dokumente, Web-Seite
- Hypertext ist Text, der durch Links ergänzt wird
- Link ist Verweis auf andere Textstelle oder Dokument (Objekt). Link kann auf selbes Objekt, Objekt im selben Rechner, oder Objekt im Netz verweisen.
- Hypermedia enthält zusätzlich zu Hypertext multimediale Anteile (Grafik, Video, Sprache), beschrieben mittels HTML (Hypertext Markup Language) oder Erweiterungen
- Referenzierung von Objekten

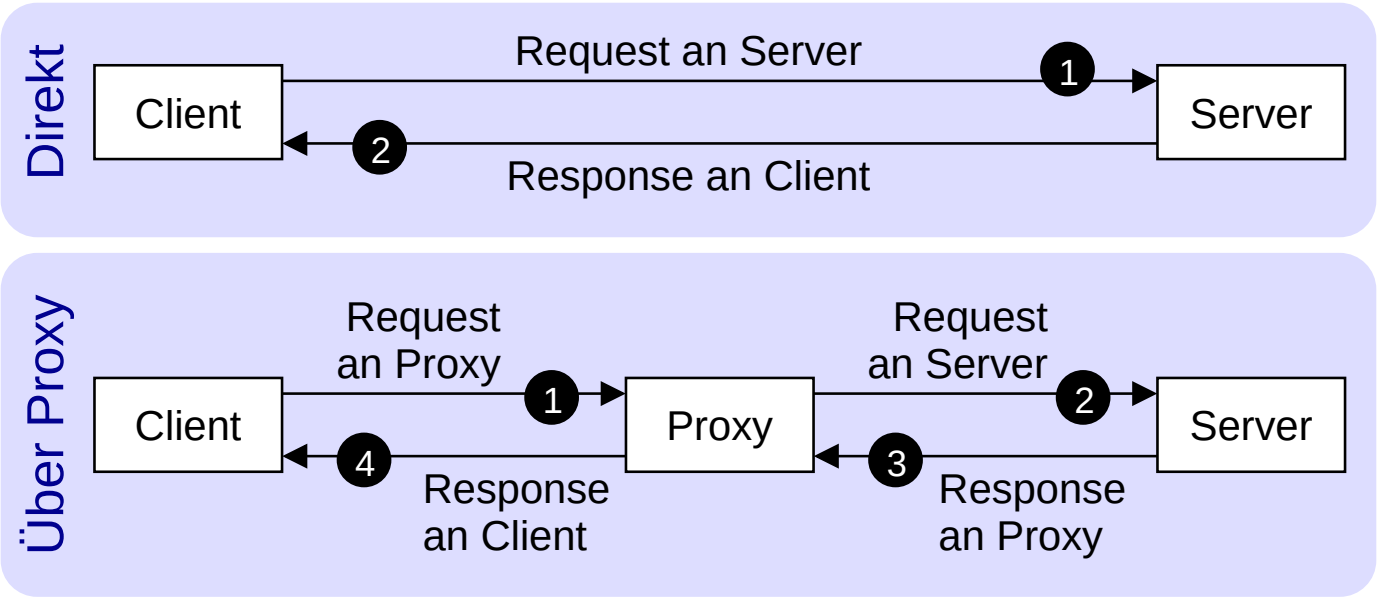
URL (Uniform Resource Locator) RFC 1738: Lagerort eines Web-Dokuments durch Serverangabe und Pfadbezeichnung

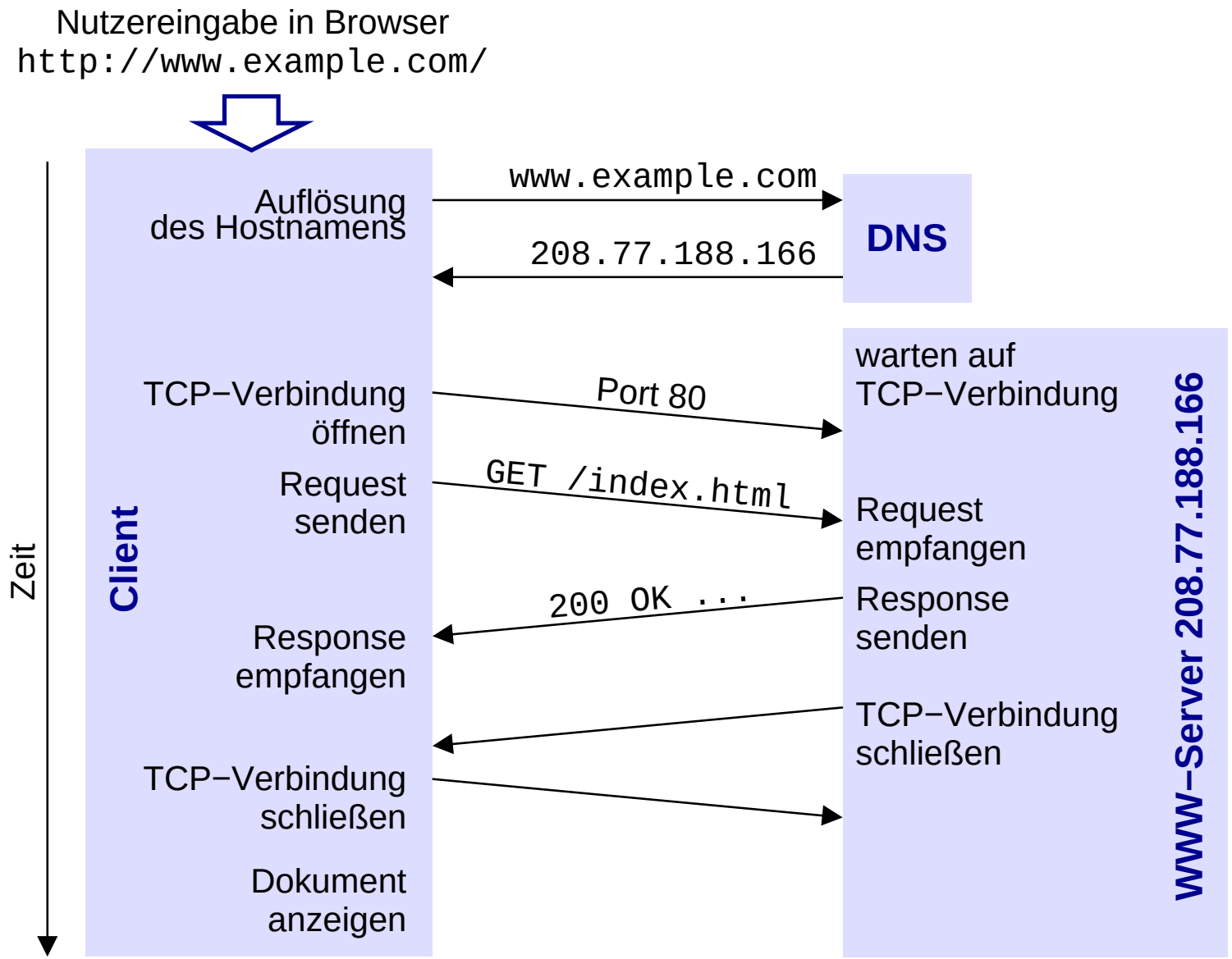
URN (Universal Resource Name): global eindeutiger langlebiger logischer Name für Objekt (ohne Lagerort)

URI (Universal Resource Identifier) RFC 1630: Objektbegriff für URL und URN

- Beschreibungssprache für Web-Dokumente
  - Standardisiert als **W3C Recommendation**
  - bietet Markup (Tags) in Bezug auf Struktur und Layout
  - viele HTML-Versionen existieren, derzeit HTML 4.01 bzw. XHTML
  - weitere Markup-Sprachen: SGML, XML, SVG
- HTML-Dokumente
  - bestehen aus Header (Titel, Formatierung, Metadaten) und Body (Inhalt)
  - Neben Fließtext können Tabellen (tables) und Formulare (forms) verwendet werden.
  - Links durch Anker (anchor) angegeben, die eine URL enthalten.
  - mehr als 80 Tags (grundlegende, in Header, für Tabellen, Dokumentteiler, Textformatierungen, Links, Bilder, Formulare, Listen, Frameelement)
  - CSS (Cascading Style Sheet): zur Präzisierung von Formatvorgaben
- HTTP-URL:
  <protokoll> “://” <user> “:” <passwd> “@” <host>:<port> “/” <path>
  - z.B. `http://www.example.com/documents/index.html`
  - Optionale (und eher seltene) Felder: user, passwd, port

- Protokoll zum Transport von Anfragen/Objekten zwischen Browser, Server und Proxy (Zwischensystem)
- TCP-basiert, Port 80 (Proxy: typischerweise Port 8080)
- HTTP ist zustandslos, pull-orientiert, unterstützt bidirektionale Übertragung und Caches im Client bzw. Proxy
- HTTP/1.1 spezifiziert in RFC 2068, 2616



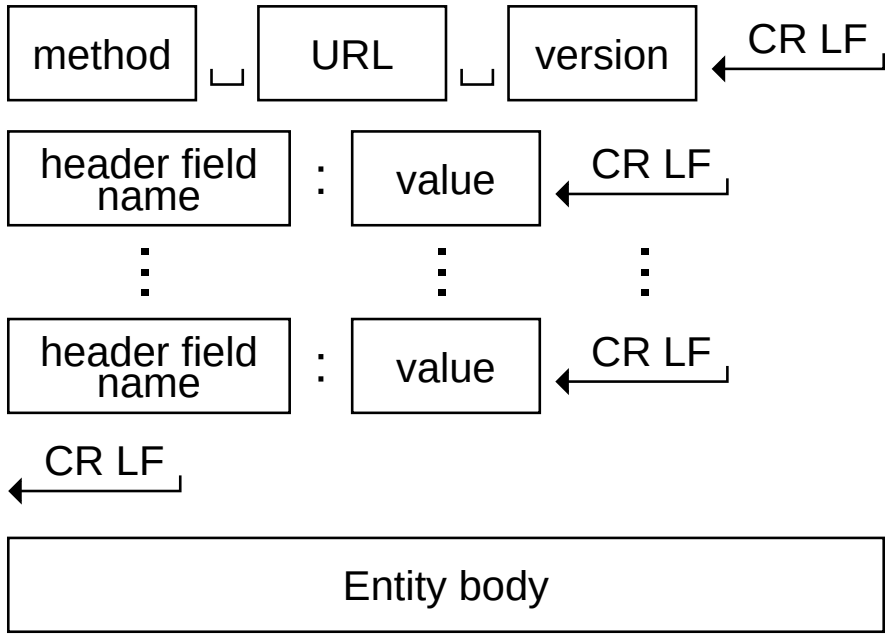

Nutzereingabe in Browser
http://www.example.com/
Zeit
Client
Auflösung des Hostnamens
www.example.com
208.77.188.166
DNS
TCP−Verbindung öffnen
Port 80
warten auf TCP−Verbindung
Request senden
GET /index.html
Request empfangen
200 OK ...
Response senden
Response empfangen
TCP−Verbindung schließen
TCP−Verbindung schließen
Dokument anzeigen
WWW−Server 208.77.188.166

Methoden

- GET: Abruf einer Datei vom Server.
- HEAD: Abruf nur von Metadaten über eine Datei.
- POST: Übertragen von Daten an den Server.
- PUT: Ablegen einer Datei auf Server
- DELETE: Löschen einer Datei auf dem Server.
- OPTIONS: Abfrage von Informationen über Kommunikationsoptionen.
- TRACE: für Testzwecke.

Statuscodes (Auswahl)

- Informational 1xx
  - 100 Continue
  - 101 Switching Protocols
- Successful 2xx
  - 200 OK
  - 206 Partial Content
- Redirection 3xx
  - 301 Moved Permanently
  - 302 Found
  - 307 Temporary Redirect
- Client Error 4xx
  - 400 Bad Request
  - 401 Unauthorized
  - 403 Forbidden
  - 404 Not Found
- Server Error 5xx
  - 500 Internal Server Error



- version: Protokollversion
- header field name: Name einer Variablen
- value: Wert einer benannten Variablen
- entity body: transportiert eine Nachricht als Teil des Dienstaufrufs

Request (Client an Server)

GET /somedir/page.html HTTP/1.1 ← Methode und Objekt
Connection: close ← Verbindung nach Response schließen
User-agent: Mozilla/4.0 ← Eigenschaften des Clients
Accept: text/html, image/gif,image/jpeg
Accept-language:fr

Response (Server an Client)

HTTP/1.1 200 OK ← Statuscode (Numerisch und Klartext
Connection: close
Date: Thu, 06 Aug 1998 12:00:15 GMT
Server: Apache/1.3.0 (Unix) ←Servertyp
Last-Modified: Mon, 22 Jun 1998 . . .
Content-Length: 6821
Content-Type: text/html ←Hinweis zum Typ des Objekts
data data data data data . . . ←Objektdaten

- Zahlreiche Webangebote (Buchungen, Verkauf, Informationsseiten)
- Statische Inhalte: Dateien (Hypertext, Multimedia) auf Dateisystem
- Dynamische Inhalte: zum Abfragezeitpunkt generiert
  - Anbindung von Datenbanksystemen und Geschäftslogik
  - Serverseitig: Programme/Skripten in dedizierten (PHP, ASP . . .) und allgemein anwendbaren Sprachen (C, Shell, Perl, Python . . .)
  - Clientseitig: Teil des Dienstes wird im Client (Browser) ausgeführt
  - Java Applets, JavaScript (Teil neuerer HTML-Standards), Flash
  - Kommunikation mit serverseitigem Skript (meist über HTTP)
- Sicherheit
  - Authentifizierung
    - IP-Adresse (nicht praktikabel)
    - Basic Authentication (Kennung/Passwort)
    - Cryptographische Zertifikate
  - HTTPS: Verschlüsselte Variante von HTTP (RFC 2660)
    - meist zusammen mit Serverauthentifizierung (Zertifikat)
    - eigentliche Übertragung über Transport Layer Security (TLS), Secure Socket Layer (SSL)(Port 443)

10.4 World Wide Web

Chunked transfer coding

- Dokumente mit (noch) unbekannter Länge (dynamischer Inhalt)
- Aufteilung in „Chunks“; Auslieferung „so, wie sie bereitstehen“

```
  1 $ telnet www.nm.ifi.lmu.de 80
[...] telnet-Nachrichten
  5 GET http://www.nm.ifi.lmu.de/rn HTTP/1.1
  6 Host: nm.ifi.lmu.de:80
  7 ↵
  8 HTTP/1.1 302 Found                ← Referenz auf anderen Ort
  9 Date: Thu, 15 Jan 2009 13:32:28 GMT
 10 Server: Apache/1.3.29 (Unix) PHP/4.3.4
 11 X-Powered-By: PHP/4.3.4
 12 Location: http://www.nm.ifi.lmu.de/teaching/Vorlesungen/2008ws/rn
 13 Transfer-Encoding: chunked       ← Länge des Dokuments noch nicht bekannt
 14 Content-Type: text/html
 15 ↵
 16 1 ↵ ← Länge des nächsten „Chunk“ (hexadezimal)
 17 ↵ ← Inhalt
 18 ↵ ← Ende des Chunks
 19 0 ↵ ← Länge des nächsten „Chunk“ = 0 heisst: fertig
 20 ↵ ← Ende des in Chunks aufgeteilten Dokuments
 21 GET http://www.nm.ifi.lmu.de/teaching/Vorlesungen/2008ws/rn/ HTTP/1.1
 22 Host: nm.ifi.lmu.de:80
 23 ↵
 24 HTTP/1.1 200 OK
 25 Date: Thu, 15 Jan 2009 13:32:57 GMT
 26 Server: Apache/1.3.29 (Unix) PHP/4.3.4
 27 X-Powered-By: PHP/4.3.4
 28 Transfer-Encoding: chunked
 29 Content-Type: text/html
 30 ↵
 31 1027  ← Länge des nächsten „Chunk“ (= 4135 dezimal)
 32
 33
 34 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">
 35 <html>
```

chunk := chunk-size CRLF
chunk-data CRLF

last-chunk := ’0’ CRLF

- Vorteile: einfach → fehlerunempfindlich
- Nachteil: viele Dienste benötigen Zustandshaltung
  - Dienstsitzung besteht aus **mehreren Schritten** (z.B. Buchung einer Fahrkarte)
  - Request/Response-Paar entspricht einem einzigen Schritt
  - Dienststatus muß **über alle Schritte** erhalten werden
- Abhilfe (zum Standard erhobene Notlösungen)
  - Statusvariablen in URL, werden per GET-Methode übertragen
  - Status wird in Daten der POST-Methode übertragen
  - Cookies: clientseitige Speicherung einer (kleinen) Datenstruktur
  - Entfremdung von HTTPS (Benutzerauthentifizierung bestimmt Sitzung)
  - SessionID: Vergabe eindeutiger Kennung für eine Dienstsitzung (oft in URL); Status serverseitig gespeichert.

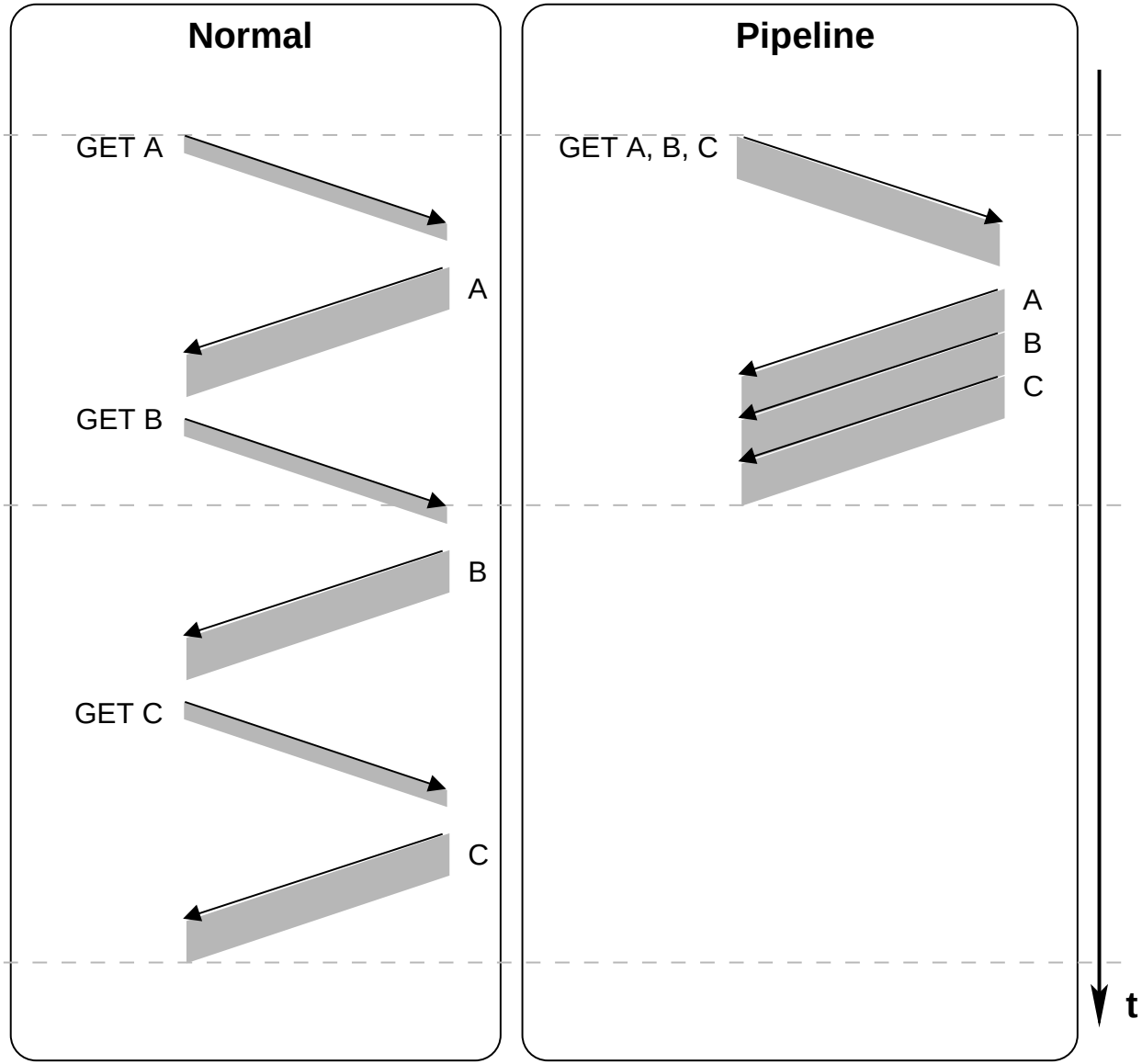
Ziele: schnelle Anzeige von Webseiten; geringe Netzlast

Einfachster Fall

- Eine TCP-Verbindung zum Server pro Request/Response-Paar
- Nach Request wird auf Response gewartet.
- Client ruft Objekt jedes Mal vollständig vom Server ab.
- Verbindung wird nach Interaktion geschlossen.

Maßnahmen zur Leistungssteigerung

- Mehrere gleichzeitige Verbindungen: Parallelisierung der Anfragen
- Persistente Verbindung: Verbindung wird für mehrere Anfragen wieder benutzt → Einsparung des Verbindungsaufbaus (“Connection: keepalive”)
- Pipelining: Client schickt mehrere Requests nacheinander; Server schickt entsprechende Responses.
- Caching: Client verwaltet lokales Cache jüngst abgerufener Objekte.
- Conditional GET: Objekt wird nur übertragen, falls neuer als Cache-Version.
- Caching proxy: Anfragen werden durch Proxy geleitet. Proxy verwaltet Cache → gemeinsames Nutzen des Cache


Normal
GET A
A
GET B
B
GET C
C
Pipeline
GET A, B, C
A
B
C
t

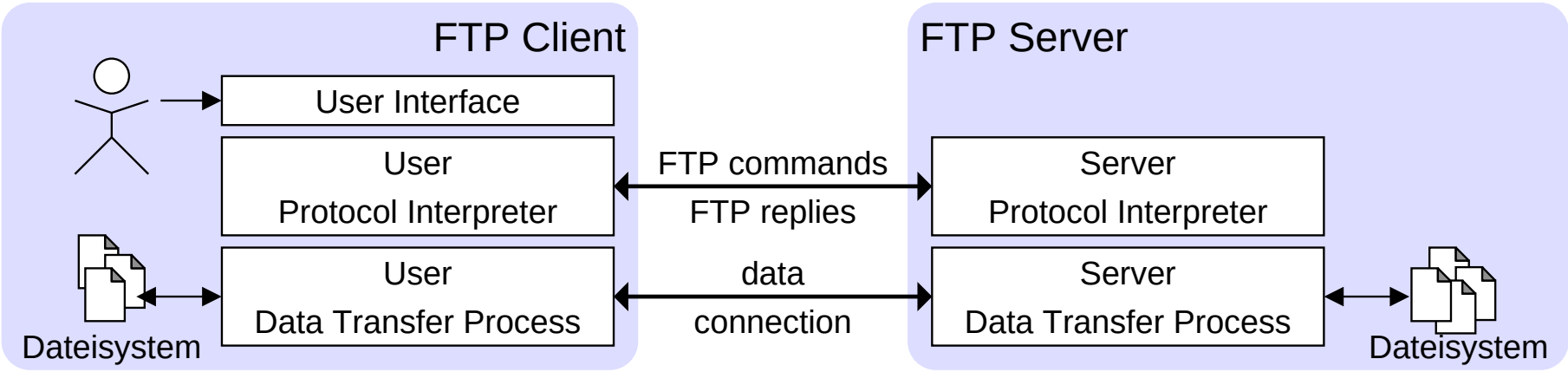
Rechnernetze und verteilte Systeme
Internet-Dienste

File Transfer Protocol

Kapitel 10.5

10.5 File Transfer Protocol

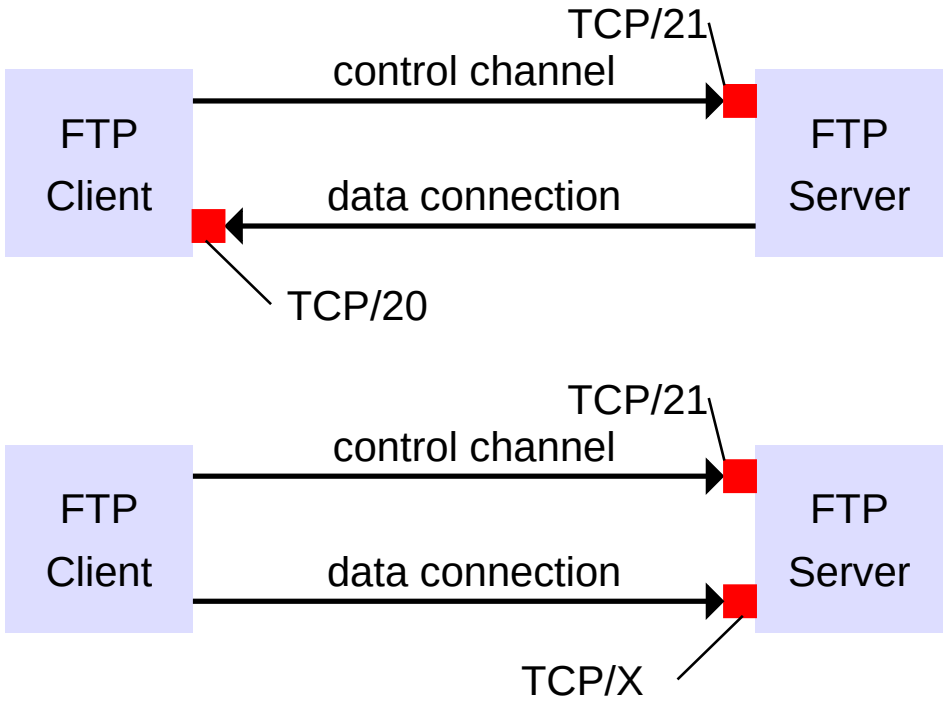
Überblick



- Übertragung von Dateien zwischen zwei Hosts (RFC 959)
- *control channel* (TCP, Port 21): Befehle, Antworten
  - out-of-band Befehle ($\rightarrow$ nicht verwechselbar mit Nutzdaten)
- *data connection* (TCP): Übertragung von Dateien (bidirektional)
  - nur für Dateiübertragung geöffnet
  - *aktives* FTP: Server öffnet data connection
  - *passives* FTP: Client öffnet data connection

10.5 File Transfer Protocol

Aktiver/Passiver Modus



- Passives FTP oft vorteilhaft in durch Paketfilter („Firewall”) geschützten Umgebungen

10.5 File Transfer Protocol

Befehle und Statuscodes (Auswahl)

MNM TEAM

Befehl: <BEFEHL> <PARAM>

- Authentifizierung
  - USER <Benutzername>
  - PASS <Paßwort>
- Verbindung
  - PORT <Nr.>: clientseitig, data channel
  - PASV: passiver Modus
  - QUIT: abmelden
- Umgang mit Dateien
  - CWD: Change Working Directory
  - MKD/RMD: Verzeichnis anlegen/löschen
  - LIST: abrufen Dateiliste
  - RETR/STOR: Datei abrufen/ablegen

Antwort: <CODE> <Text>

331 username OK, password required

125 data connection already open; transfer starting

425 can't open data connection

452 error writing file

Übertragung in ASCII über control channel

Rechnernetze und verteilte Systeme
Internet-Dienste

Internet Relay Chat

Kapitel 10.6

IRC – Internet Relay Chat

- Austausch von (Text-) Nachrichten zwischen Nutzern (“text-based conferencing”)
- Zeichenorientiert, 8-Bit/Zeichen, (US-ASCII vorgeschlagen, aber nicht vorgeschrieben)
- Client-Server Paradigma, aber auch Peer-to-Peer
- Basiert auf TCP, Ports 194, 6667

Einsatzszenarien

- Direkt durch Austausch von Textnachrichten nutzbar (telnet)
- Nutzung als Kollaborations- und Unterhaltungsmedium („chat”)
  - Anforderungen an Darstellung (Farbe, Textdarstellung etc)

  $\Rightarrow$ entsprechende Protokollelemente
  - Erweiterungen z.B. Übertragung von Whiteboard-Daten
- Nutzung als Angelpunkt für Kommunikation in Verteilten Systemen (manchmal für unerwünschte Aktivitäten)

Standardisierung, Implementierung, Erweiterungen

- Frei entstanden, aber von der IETF in RFCs beschrieben: RFC 2810 (Architektur), RFC 2813 (Serversicht), RFC 2812 (Clientsicht), RFC 2811 (Kanalverwaltung)
- zahlreiche Implementierungen von Clients (text-basiert z.B. Epic, BitchX, graphisch z.B. mIRC) sowie Serversoftware
- Variante mit verschlüsselter Übertragung (IRC over SSL, IRCS), Port 994

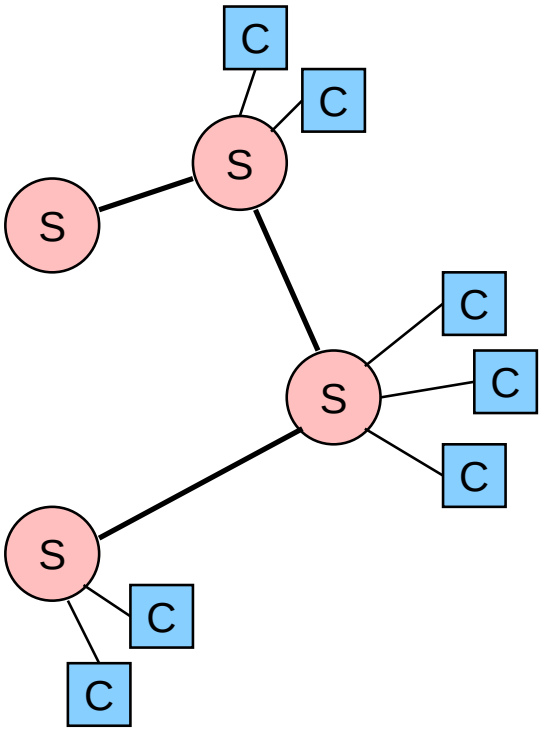
- Laufende Entwicklung spezieller Erweiterungen

$\Rightarrow$ gemeinsames Protokoll-Subset

$\Rightarrow$ diverse weitere Funktionen in Abhängigkeit von Softwarepaket

10.6 Internet Relay Chat
Architektur

- Server
  - In Baumtopologie organisiert (*spanning tree*)
  - Verbunden mit anderen Servern sowie mit Clients
  - Globale Sicht (jeder Server „kennt“ alle anderen Server)
- Client: Teilnehmer, der nicht ein Server ist
- Proxy: Statthalter für Client(s)



- Nicht nur ein IRC-Netz
- Beispiele:
  - EFnet
  - IRCnet
  - QuakeNet
  - Undernet
  - Freenode
  - DALnet

IRC-Netz überlagert IP-Netz (sog. „overlay“)

Grundkonzepte

- Kommunikationspartner: Server, Client, Proxy
- Kanal (channel): Multicast-Gruppe für Textnachrichten (Serververwaltet, IRC-weit)
- Liste: Multicast-Gruppe (Nutzerdefiniert)
- Server wählen korrekte Richtung im Baum ⇒kürzester Pfad

1:1 Unicast
- Vermittlung von Nachrichten zwischen zwei Clients
- Direkter Austausch zwischen Clients (Peer-to-Peer)

1:n Multicast
- an Kanal oder Liste
- anhand Host/Server Maske

1:* Broadcast zu allen Servern und Clients

- Kanal: realisiert die Funktion einer Diskussionsgruppe.
- Referenziert mit `Kanalname := prefix name`
  `prefix := ('&' | '#' | '+' | '!')`
  `name` : bis zu 50 Zeichen, Groß-/Kleinschreibung egal
- Beispiel: `#blafasel`
- Prefixe erzeugen 4 versch. Namensräume;
  - `name` ist innerhalb eines Namensraumes eindeutig
  - `'&'`: Kanal ist lokal bzgl. eines Servers (Namensraum auf Server beschränkt)
  - `'!'`: Sog. *safe channel*;
  - `'#', '+'`: Standardtypen für Kanäle (`'+'` = keine Modi)
- Flags: Parameter, die für Kanal (z.T. bei dessen Erzeugung) durch Operatöre gesetzt werden. Z.B: `'m'` = moderierter K.; `'a'` = Clients werden anonymisiert; `'i'` = *"invite only"*
- Maske (*channel mask*): regulärer Ausdruck zur Auswahl von Kanälen

- Idee: Austausch von Nachrichten unabhängig von IRC-Server-Netz
- $\rightarrow$Erweiterung des IRC um Peer-to-Peer Funktionalität
- Grundlage: Direct Client-to-Client (DCC)
- DCC-Aushandlung zwischen 2 Clients über einen IRC-Kanal
  - Passiver Client: lauscht auf TCP-Port, teilt IP-Adr./Portnr. mit
  - Aktiver Client: initiiert TCP-Verbindung
- Austausch von Textnachrichten (DCC Chat)
- Übertragung von Dateien (DCC Send)

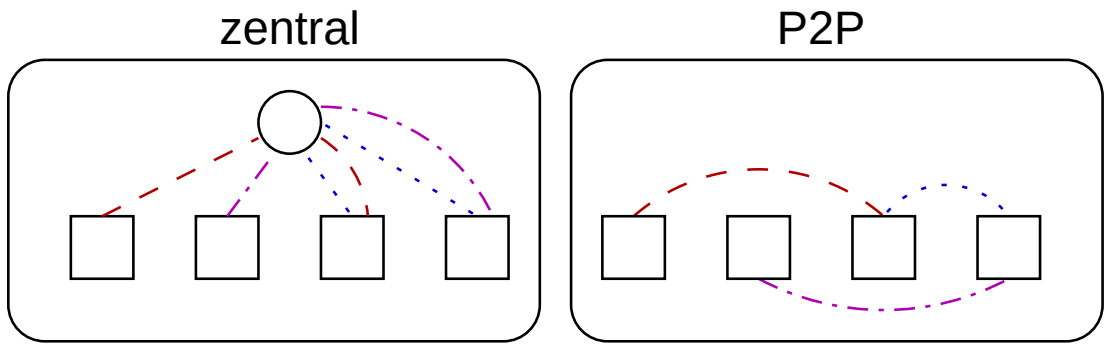
Rechnernetze und verteilte Systeme
Internet-Dienste

Peer-to-Peer Dienste

Kapitel 10.7

Motivation

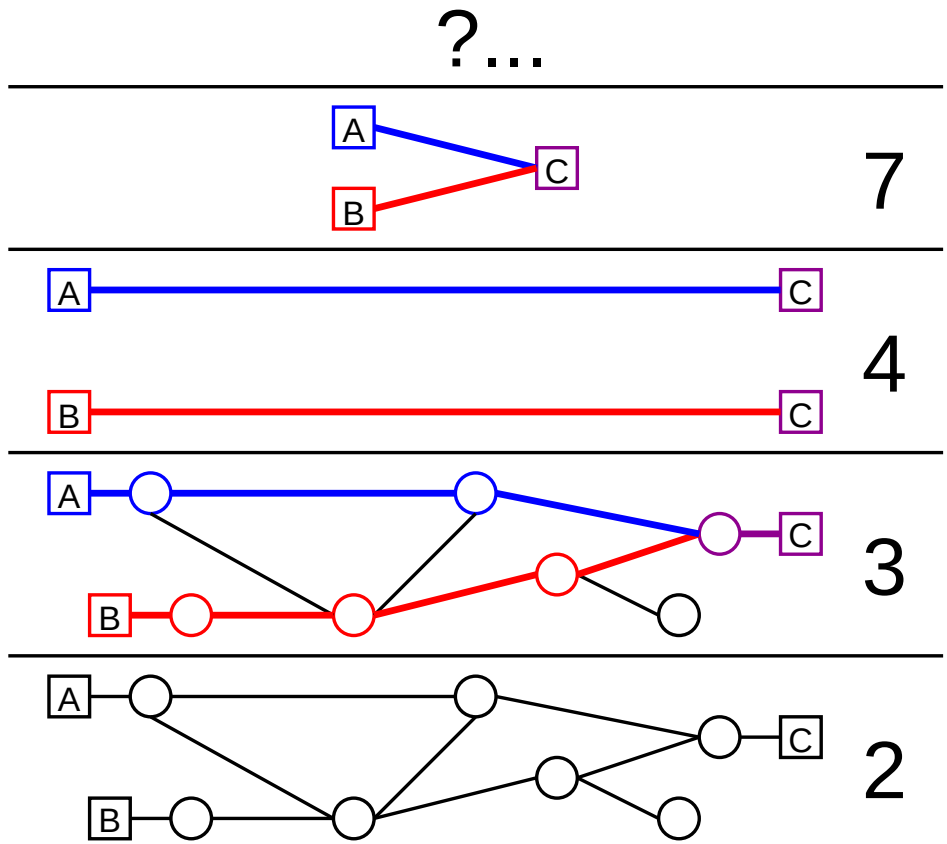
- Direkte Kommunikation zwischen Dienstnutzern
- Bessere Skalierbarkeit durch Verzicht auf zentrale Server
- Dimensionierung der Transportnetze abhängig von Verkehrsaufkommen



Nutzerdienste

- Filesharing (Napster, Gnutella, Bittorrent, . . .)
- Chat (proprietäre Dienste ICQ, AIM, . . .)
- Spiele

- Vernetzung von Entitäten der Anwendungsschicht zu bestimmten Zweck
- Realisierung von Vermittlungs- und Transportfunktionen
- Nutzung der Internet-Transportschicht als Grundlage



- Mit zentraler Einrichtung
  - Zentraler Server als Index für Knoten und Objekte
  - z.B. Napster: Suche in zentralem Index; Datenübertragung zwischen Clients
  - Vorteile: Konsistenz, u.U. geringerer Ressourcenverbrauch
  - Nachteile: Skalierbarkeit, *Single-Point-of-Failure*
- Reines Peer-to-Peer
  - Verzicht auf jegliche zentrale Einrichtung
  - Signalisierung sowie Nutzfunktionen nur zwischen gleichberechtigten Knoten
  - z.B. Gnutella 0.4
  - Vorteile: Skalierbarkeit, Ausfallsicherheit
  - Nachteile: größerer Signalisierungsaufwand
- Hybride
  - Idee: Einführung temporärer, lokaler Hierarchie
  - Zweck: Leistungsvorteile, ohne dezentrale Architektur aufzugeben
  - Beispiele: Gnutella 0.6, Bittorrent

Eigenschaften des Gnutella-Netzes

- Reines Peer-to-Peer Netz; keine zentralen Instanzen (Version 0.4)
- Zweck: Austausch einzelner Dateien zwischen Gnutella-Knoten
- Knoten: sog. Servent (**Ser**ver Cli**ent**)
- Adressierung mit GUIDs im Overlay-Netz; mit IP-Adressen zum Internet hin
- Jeder Knoten hält Verbindungen zu ca. 3 anderen Knoten

Signalisierung

- GNUTELLA CONNECT: Verbindungsaufbau über TCP-Verbindung
- PING: zur Entdeckung des Netzes; durch Flooding vermittelt
- PONG: Antwort auf PING; enthält IP-Adressen weiterer Knoten
- QUERY: Suche nach einer Datei
- QUERY-HIT: Antwort(en) auf QUERY

Fragen zu Kapitel 10

- Ohne welche zusätzliche Internetanwendung funktionieren Mail, Filetransfer, Web nicht?
- Worin unterscheidet sich die Nutzung der Socket-Dienstschnittstelle für Anwendungen bei der Nutzung von TCP oder UDP
- Welche Haupt-Bausteine machen ein Email-System aus? Welche Protokolle werden zwischen den Bausteinen benutzt?
- Welche Hauptkomponenten machen ein Web-System aus?
- Wie ist eine URL aufgebaut?
- Realisieren SMTP bzw. HTTP ein Push oder Pull-Modell?
- Jeder Internet-Host hat mindestens einen lokalen und einen autorisierten Name-Server. Welche Rolle spielt jeder dieser Server im DNS?