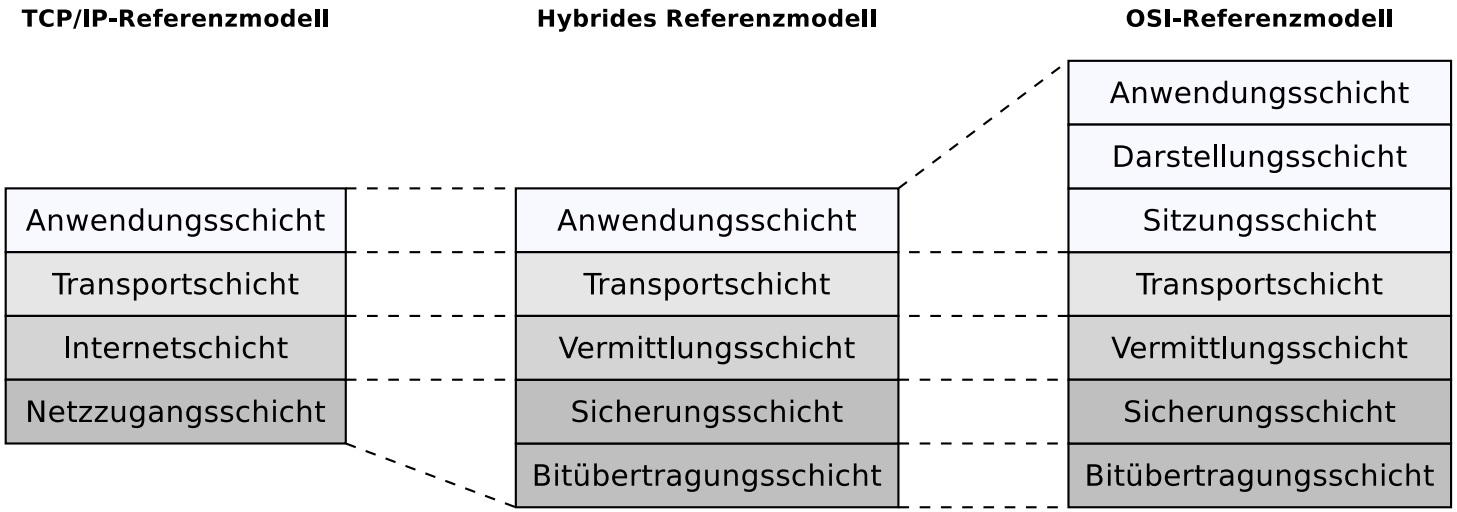
12. Foliensatz
Betriebssysteme und Rechnernetze

Prof. Dr. Christian Baun

Frankfurt University of Applied Sciences
(1971–2014: Fachhochschule Frankfurt am Main)
Fachbereich Informatik und Ingenieurwissenschaften
christianbaun@fb2.fra-uas.de

Anwendungsschicht

- Enthält alle Protokolle, die mit Anwendungsprogrammen (zum Beispiel Browser oder Email-Programm) zusammenarbeiten
- Hier befinden sich die eigentlichen Nachrichten (zum Beispiel HTML-Seiten oder Emails) entsprechend dem jeweiligen Anwendungsprotokoll



Übungsblatt 5 wiederholt die für die Lernziele relevanten Inhalte dieses Foliensatzes

- Geräte: keine
- Protokolle: DNS, DHCP, NTP, Telnet, SSH, HTTP, SMTP, FTP...

Sinnvolle Themen zur Anwendungsschicht...

- Anwendungsprotokolle
  - Namensauflösung (DNS)
  - Automatische Vergabe von Adressen (DHCP)
  - Zeitsynchronisierung (NTP)
  - Fernsteuerung von Computern (Telnet, SSH)
  - Übertragung von Daten (HTTP)
  - Emails austauschen (SMTP)
  - Emails herunterladen (POP3)
  - ~~Dateien hochladen und herunterladen (FTP)~~

Domain Name System (DNS)

- Protokoll zur **Namensauflösung** von Domain-Namen zu IP-Adressen

RFC 1034 and 1035

- Analog zur Telefonauskunft
  - Person/Familie/Firma $\Longrightarrow$ Telefonnummer
  - Rechnername/Website $\Longrightarrow$ IP-Adresse

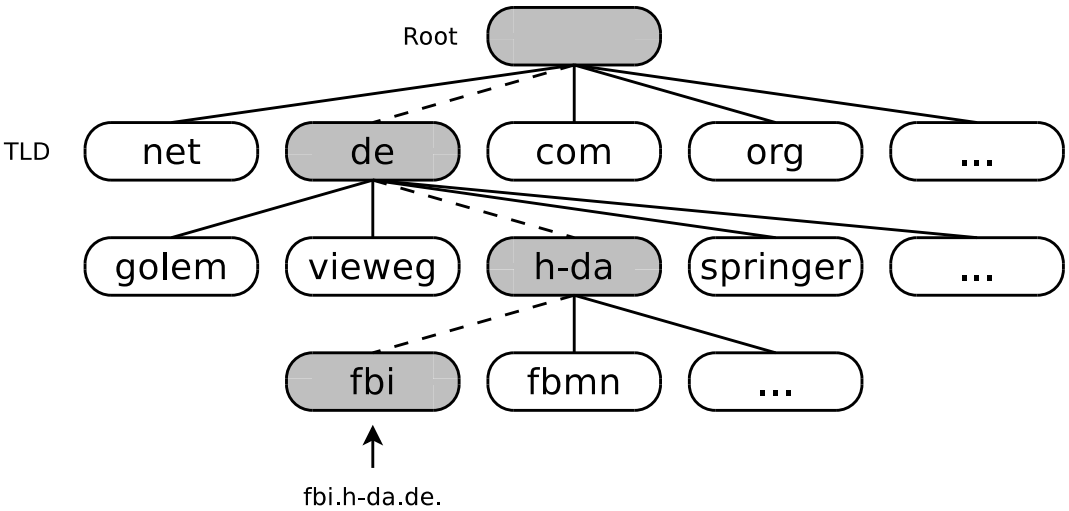
Entwicklung 1983 von Paul Mockapetris

- DNS löste die lokalen Namenstabellen in der Datei `/etc/hosts` ab, die bis dahin für die Verwaltung der Namen/Adressen-Zuordnungen zuständig waren
  - Diese waren der zunehmenden Zahl von Neueinträgen nicht mehr gewachsen
- Basiert auf einem hierarchischen Namensraum
  - Die Information mit den Zuordnungen sind in separate Teile gegliedert und im gesamten Internet auf **Nameservern** verteilt

Domain-Namensraum (1/2)

- Der Domain-Namensraum hat eine **baumförmige Struktur**
  - Die Blätter und Knoten heißen **Labels**
  - Jeder Unterbaum ist eine **Domäne**
- Ein vollständiger Domainname besteht aus der Verkettung aller Labels eines Pfades
- Label sind alphanumerische Zeichenketten
  - Als einziges Sonderzeichen ist der Bindestrich erlaubt
  - Labels sind 1 bis 63 Zeichen lang
  - Labels dürfen nicht mit einem Bindestrich anfangen oder enden
  - Jedes Labels endet mit einem Punkt
- Domainnamen werden mit einem Punkt abgeschlossen
  - Wird meist weggelassen, gehört rein formal aber zu einem vollständigen Domainnamen – **Fully Qualified Domain-Name** (FQDN) dazu
- Ein vollständiger Domainname ist z.B. `www.h-da.de.`

Domain-Namensraum (2/2)



- Domainnamen werden von rechts nach links aufgelöst
  - Je weiter rechts ein Label steht, umso höher steht es im Baum

- Die erste Ebene unterhalb der Wurzel heißt **Top-Level-Domain** (TLD)
- Die DNS-Objekte einer Domäne (zum Beispiel die Rechnernamen) werden als Satz von **Resource Records** (RR) in einer Zonendatei gehalten, die auf einem oder mehreren Nameservern vorhanden ist
- Die Zonendatei heißt häufig einfach **Zone**

Root-Nameserver

http://www.root-servers.org (Stand: Mai 2020)

- Die 13 Root-Nameserver (`A` bis `M`) publizieren die **Root-Zone** des DNS
  - Deren Domain-Namen haben die Form `buchstabe.root-servers.net`
  - Die Root-Zone enthält ca. 3000 Einträge und ist die Wurzel des DNS
    - Sie enthält die Namen und IPs der für die TLDs zuständigen Nameserver
- Die Root-Server bestehen nicht aus einem, sondern mehreren physischen Servern, die zu einem logischen Server verbunden sind
  - Diese Rechner befinden sich an verschiedenen Standorten weltweit und sind via **Anycast** über dieselbe IP-Adresse erreichbar

Name	IPv4-Adresse	IPv6-Adresse	Ort	Standorte	Betreiber
A	198.41.0.4	2001:503:ba3e::2:30	verteilt (Anycast)	53	Verisign, Inc.
B	199.9.14.201	2001:500:200::b	verteilt (Anycast)	6	Information Sciences Institute
C	192.33.4.12	2001:500:2::c	verteilt (Anycast)	10	Cogent Communications
D	199.7.91.13	2001:500:2d::d	verteilt (Anycast)	156	University of Maryland
E	192.203.230.10	2001:500:a8::e	verteilt (Anycast)	308	NASA Ames Research Center
F	192.5.5.241	2001:500:2f::f	verteilt (Anycast)	252	Internet Systems Consortium, Inc.
G	192.112.36.4	2001:500:12::d0d	verteilt (Anycast)	6	Defense Information Systems Agency
H	198.97.190.53	2001:500:1::53	verteilt (Anycast)	8	U.S. Army Research Lab
I	192.36.148.17	2001:7fe::53	verteilt (Anycast)	72	Netnod
J	192.58.128.30	2001:503:c27::2:30	verteilt (Anycast)	185	Verisign, Inc.
K	193.0.14.129	2001:7fd::1	verteilt (Anycast)	77	RIPE NCC
L	199.7.83.42	2001:500:9f::42	verteilt (Anycast)	165	ICANN
M	202.12.27.33	2001:dc3::35	verteilt (Anycast)	9	WIDE Project

Aufbau der DNS-Datenbank und Ressourceneinträge

Sie wissen bereits...

- DNS ist eine Art verteilte Datenbank mit baumförmiger Struktur
- Beim Internet-DNS liegen die Daten auf einer Vielzahl weltweit verteilter Server, die untereinander über Verweise (*Delegierungen*) verknüpft sind
- In jedem Nameserver existieren ≥ 1 Zonendateien

- Die Zonendateien enthalten Listen von **Resource Records** (RR)
- Jeder RR („*Ressourceneintrag*“) besteht aus 5 Elementen
  `<Name, Wert, Typ, Klasse, TTL>`
- Die Tabelle enthält einige Typen von RRs

Typ	Beschreibung
`NS`	Definiert, welcher Nameserver für die Zone zuständig ist oder verknüpft Zonen zu einem Zonen-Baum (Delegation)
`A`	Enthält die IPv4-Adresse eines Hosts
`AAAA`	Enthält die IPv6-Adresse eines Hosts
`SOA`	Enthält Angaben zur Verwaltung der Zone wie den Namen und die Email-Adresse des Administrators
`CNAME`	Liefert einen Alias-Domain-Namen für einen bestimmten Host
`MX`	Weist einem Namen einen SMTP-Mailserver zu. Alle anderen Dienste nutzen `CNAME`, `A` und `AAAA` Resource Records für die Namensauflösung
`PTR`	Weist einer IP-Adresse einen oder mehrere Hostname(s) zu. Gegenstück zur üblichen Zuordnung einer oder mehrerer IPs zu einem Hostnamen per `A` oder `AAAA` Resource Record

Beispiel einer Namensauflösung (1/5)

- Im folgenden Beispiel wird der Namen `www.fh-frankfurt.de.` mit dem Kommandozeilenwerkzeug `dig` aufgelöst

```
dig +trace +additional -t A www.fh-frankfurt.de.
```

- `-t A` $\Longrightarrow$ `A` Resource Record (die IPv4-Adresse) anfragen
- `+trace` $\Longrightarrow$ Die einzelnen Antworten auf dem Pfad durch die Nameserver-Hierarchie ausgeben
- `+additional` $\Longrightarrow$ Nameserver verwalten für Delegierungen nicht nur `NS` Resource Records, sondern teilweise auch deren IP-Adressen in Form von `A` oder `AAAA` RRs. Diese Option sorgt dafür, dass sie mit ausgeben werden

- Auf dem Weg zur IP müssen nacheinander 4 Nameserver befragt werden

Auf den folgenden Folien befinden sich in der Ausgabe von `dig` auch mehrere DNSSEC-Resource Records (RR). DNSSEC bietet Authentizität und Integrität der DNS-Daten

- `RRSIG` = Signature Resource Record = Signatur (Digitale Unterschrift) eines DNS-Resource-Record-Sets
- `NSEC3` = Gehashter nächster sicherer Eintrag in der Zone (*Chain-of-trust*)
- `DS` = Delegation Signer = Dient der Verkettung von DNSSEC-signierten Zonen. Somit werden mehrere DNS-Zonen zu einer Chain-of-trust zusammengefasst und können über einen einzigen öffentlichen Schlüssel validiert werden

Beispiel einer Namensauflösung (2/5)

```
$ dig +trace +additional -t A www.fh-frankfurt.de.

; <<>> DiG 9.10.3-P4-Debian <<>> +trace +additional -t A www.fh-frankfurt.de.
;; global options: +cmd
.                       515463  IN      NS      a.root-servers.net.
.                       515463  IN      NS      b.root-servers.net.
.                       515463  IN      NS      c.root-servers.net.
.                       515463  IN      NS      d.root-servers.net.
.                       515463  IN      NS      e.root-servers.net.
.                       515463  IN      NS      f.root-servers.net.
.                       515463  IN      NS      g.root-servers.net.
.                       515463  IN      NS      h.root-servers.net.
.                       515463  IN      NS      i.root-servers.net.
.                       515463  IN      NS      j.root-servers.net.
.                       515463  IN      NS      k.root-servers.net.
.                       515463  IN      NS      l.root-servers.net.
.                       515463  IN      NS      m.root-servers.net.
.                       515463  IN      RRSIG   NS 8 0 518400 20200602050000 2020052...
;; Received 525 bytes from 10.0.0.2#53(10.0.0.2) in 12 ms
```

- 10.0.0.2 (letzten Zeile) ist der Nameserver des abfragenden Rechners
  - Dieser Nameserver kennt die IP-Adressen der Root-Nameserver
  - Die Adressen der Root-Nameserver ändern sich selten und müssen allen Nameservern bekannt sein

Beispiel einer Namensauflösung (3/5)

```
de.                     172800  IN      NS      s.de.net.
de.                     172800  IN      NS      n.de.net.
de.                     172800  IN      NS      a.nic.de.
de.                     172800  IN      NS      f.nic.de.
de.                     172800  IN      NS      l.de.net.
de.                     172800  IN      NS      z.nic.de.
de.                     86400   IN      DS      45580 8 2 918C32E2F12211766...
s.de.net.               172800  IN      A       195.243.137.26
s.de.net.               172800  IN      AAAA    2003:8:14::53
n.de.net.               172800  IN      A       194.146.107.6
n.de.net.               172800  IN      AAAA    2001:67c:1011:1::53
a.nic.de.               172800  IN      A       194.0.0.53
a.nic.de.               172800  IN      AAAA    2001:678:2::53
f.nic.de.               172800  IN      A       81.91.164.5
f.nic.de.               172800  IN      AAAA    2a02:568:0:2::53
l.de.net.               172800  IN      A       77.67.63.105
l.de.net.               172800  IN      AAAA    2001:668:1f:11::105
z.nic.de.               172800  IN      A       194.246.96.1
z.nic.de.               172800  IN      AAAA    2a02:568:fe02::de
;; Received 753 bytes from 198.41.0.4#53(a.root-servers.net) in 24 ms
```

- Aus den 13 Root-Nameservern wurde zufällig `a.root-servers.net` ausgewählt, um ihm die Frage nach `www.fh-frankfurt.de.` zu stellen
- Die Antwort enthält 6 Nameserver zur Auswahl, die für die Zone `de.` verantwortlich sind
  - Bei allen Servern ist die Abfrage auch mittels IPv6 (`AAAA`) möglich

Beispiel einer Namensauflösung (4/5)

```
fh-frankfurt.de.          86400   IN      NS      deneb.dfn.de.
fh-frankfurt.de.          86400   IN      NS      medusa.fh-frankfurt.de.
tjlb7qboj...s1lg16.de. 7200 IN NSEC3 1 1 15 CA12B74...R67IU NS SOA RRSIG DNSKEY NSEC3PARAM
ck6ochdub...5a0eut.de. 7200 IN NSEC3 1 1 15 CA12B74...KPHCB A RRSIG
tjlb7qboj...s1lg16.de. 7200 IN RRSIG NSEC3 8 2 7200 20200528085231 2020051...
ck6ochdub...5a0eut.de. 7200 IN RRSIG NSEC3 8 2 7200 20200528095239 2020051...
medusa.fh-frankfurt.de. 86400   IN      A       192.109.234.209
deneb.dfn.de.           86400   IN      A       192.76.176.9
;; Received 637 bytes from 77.67.63.105#53(l.de.net) in 23 ms
```

- Aus den 6 genannten Nameservern wurde zufällig `l.de.net` ausgewählt, um ihm die Frage nach `www.fh-frankfurt.de.` zu stellen
- Die Antwort enthält 2 möglichen Delegierungen (`deneb.dfn.de.` und `medusa.fh-frankfurt.de.`) zur Auswahl, die für die Zone `fh-frankfurt.` verantwortlich sind

Beispiel einer Namensauflösung (5/5)

```
www.fh-frankfurt.de.      86400   IN      CNAME   squid01.dv.fh-frankfurt.de.
squid01.dv.fh-frankfurt.de. 86400 IN      A       192.109.234.216
fh-frankfurt.de.          86400   IN      NS      deneb.dfn.de.
fh-frankfurt.de.          86400   IN      NS      medusa.fh-frankfurt.de.
deneb.dfn.de.             86400   IN      A       192.76.176.9
medusa.fh-frankfurt.de.   86400   IN      A       192.109.234.209
;; Received 166 bytes from 192.76.176.9#53(deneb.dfn.de) in 23 ms
```

- Aus den 2 genannten Nameservern wurde zufällig `deneb.dfn.de` ausgewählt, um ihm die Frage nach `www.fh-frankfurt.de.` zu stellen
- `www.fh-frankfurt.de.` ist nur ein Alias (`CNAME`) für `squid01.dv.fh-frankfurt.de.`
- Ergebnis: Die IP von `www.fh-frankfurt.de.` bzw. `squid01.dv.fh-frankfurt.de.` ist 192.109.234.216

Protokoll von DNS

- DNS-Anfragen werden meist per UDP Port 53 zum Namensserver gesendet
- Die maximal zulässige Länge einer DNS-Antwort via UDP beträgt 512 Bytes
- Längere DNS-Antworten sendet der Nameserver via TCP

Dynamic Host Configuration Protocol (DHCP)

- Ermöglicht die Zuweisung der Netzwerkkonfiguration (IP-Adresse, Netzmaske, Default-Gateway, Nameserver, usw.) an Netzwerkgeräte mit Hilfe eines **DHCP-Clients** durch einen **DHCP-Server**
  - Speziell bei mobilen Geräten ist es nicht sinnvoll, feste IPs zu vergeben
    - Bei Änderungen an der Topologie des Netzes müsste man ansonsten auf allen Clients die Netzwerkeinstellungen anpassen
    - Bei DHCP wird nur die Konfiguration des DHCP-Servers angepasst
- Verwendet UDP via Ports 67 (Server oder Relay-Agent) und 68 (Client)

RFC 2131

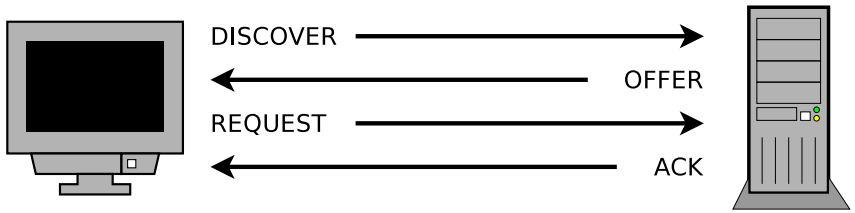
- Ein DHCP-Server verfügt über einen **Pool an IPs** und verteilt diese an Clients
- Damit ein DHCP-Client einen DHCP-Server nutzen kann, muss sich dieser **im selben logischen Netz** befinden
  - Grund: DHCP verwendet **Broadcasts** und Router leiten diese nicht weiter

Liegt der DHCP-Server in einem anderen logischen Netz, muss ein **DHCP-Relay** die Anfragen an den DHCP Server weiterleiten

Arbeitsweise von DHCP (1/2)

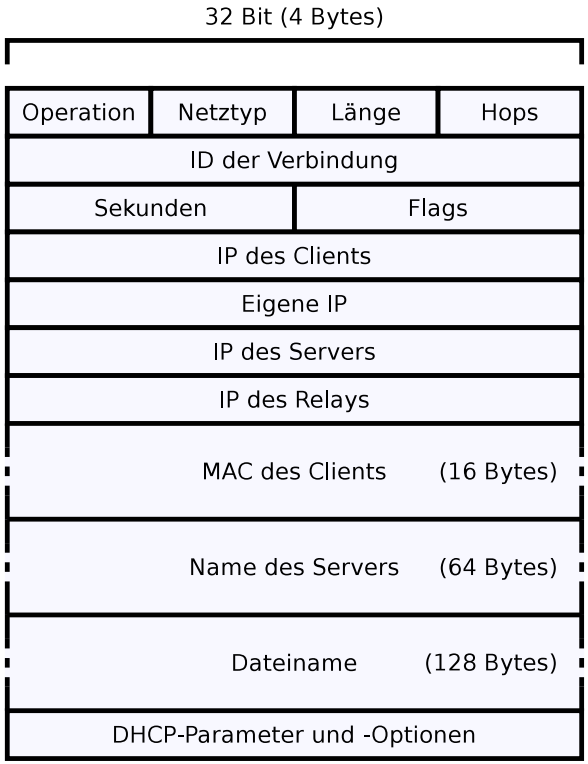
1. Ein Client ohne IP-Adresse sendet als **Broadcast** eine Anfrage (**DHCP-Discover**) an die erreichbaren DHCP-Server
   - Die Absender-IP-Adresse des Broadcast ist 0.0.0.0 und die Zieladresse ist 255.255.255.255
2. Jeder erreichbare DHCP-Server mit freien IP-Adressen in seinem Pool antwortet auf die Anfrage mit einem Adressangebot (**DHCP-Offer**)
   - Das Adressangebot wird als **Broadcast** (Zieladresse 255.255.255.255) oder **Unicast** (an die angebotene IP-Adresse) gesendet
   - Ob Broadcast oder Unicast hängt davon ab, ob der Client das Broadcast-Bit in der DHCP Discover-Nachricht gesetzt hat
3. Der DHCP-Client nimmt ein Adressangebot an, indem er eine Anfrage (**DHCP-Request**) via **Broadcast** ins Netzwerk schickt
   - Die Nachricht enthält die ID des gewünschten DHCP-Servers
   - Eventuell vorhandene weitere DHCP-Server erkennen in der Broadcast-Nachricht die Absage für ihre Adressangebote
4. Der Server bestätigt die Adressanfrage mit **DHCP-Ack** via **Broadcast** oder **Unicast** und markiert die IP in seinem Adresspool als vergeben
   - Ob Broadcast oder Unicast hängt davon ab, ob der Client das Broadcast-Bit in der DHCP Discover-Nachricht gesetzt hat
   - Er kann die Anfrage auch mit **DHCP-Nak** ablehnen

Arbeitsweise von DHCP (2/2)



- Hat ein DHCP-Server eine Adresse vergeben und dies mit **DHCP-Ack** bestätigt, trägt er in seiner Datenbank bei der Adresse ein *Lease* ein
  - Sind alle Adressen vergeben (verliehen), können keine weiteren Clients mit IP-Adressen versorgt werden
- Jede Adresse besitzt ein Verfallsdatum (*Lease Time*)
  - Dieses wird mit der Bestätigung (**DHCP-Ack**) an den Client übermittelt
  - Aktive Clients verlängern den Lease regelmäßig nach der Hälfte der Lease-Zeit mit einem erneuten **DHCP-Request** direkt via **Unicast** an den Server (nicht per Broadcast)
  - Der Server antwortet mit einer erneuten Bestätigung (**DHCP-Ack**) mit den identischen Daten wie vorher und einem neuen Verfallsdatum
  - Ist das Verfallsdatum abgelaufen, kann der Server die Adresse bei Anfragen neu vergeben

Aufbau von DHCP-Nachrichten



- **Operation** legt fest, um was für eine DHCP-Nachricht es sich handelt
  - 1 = Anforderung (*Request*) eines Clients
  - 2 = Antwort (*Reply*) eines Servers
- **Netztyp** gibt die Vernetzungstechnologie an
  - 1 = Ethernet, 6 = WLAN
- **Länge** definiert die Länge der physischen Netzadresse in Bytes
- **Hops** ist optional und gibt die Anzahl der DHCP-Relays auf dem Pfad an
- **Flags** hier ist das Broadcast-Flag
  - Es gibt an, ob DHCP-Offer und DHCP-ACK via Broadcast oder Unicast gesendet werden
- **Dateiname** ist optional und enthält den Namen einer Datei, die sich der Client via Trivial File Transfer Protocol (TFTP) holen soll
  - Damit kann ein Endgerät über das Netzwerk booten

Network Time Protocol (NTP)

- Standard zur Synchronisierung von Uhren zwischen Computersystemen

RFC 5905 beschreibt das Protokoll und die Algorithmen im Detail

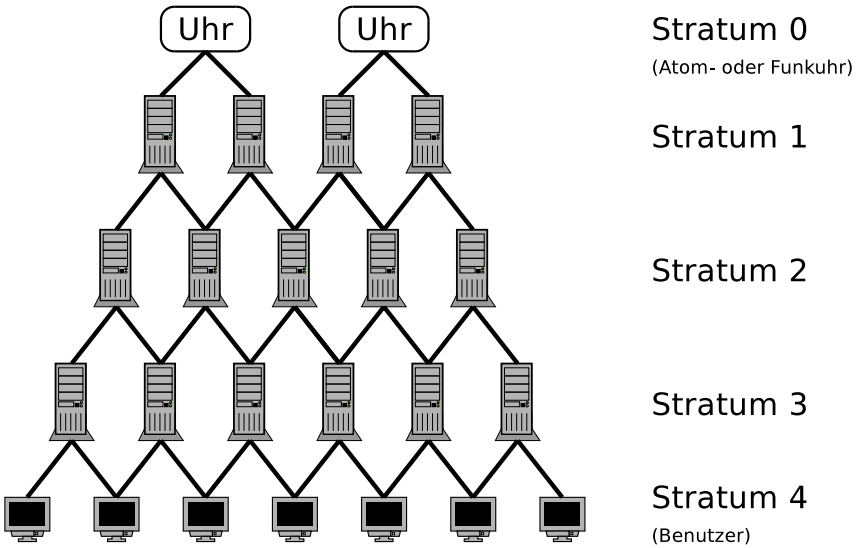
- NTP steht für das Protokoll und für die Referenzimplementierung
  - Verwendet UDP via Port 123

Entwickelt im 1985 von David L. Mills an der Universität von Delaware

- Die lokale Uhr wird vom lokalen Hintergrundprozess (Dämon) der NTP-Software mit einem externen Zeitsignal (z.B. Atom-Uhr, lokaler Funkempfänger oder entfernter NTP-Server via NTP) synchronisiert
- Die Zeitstempel im NTP sind 64 Bit lang
  - 32 Bit enthalten die *UNIX-Zeit* (Sekunden seit dem 1. Januar 1970 00:00:00 Uhr)
  - 32 Bit enthalten den Sekundenbruchteil
  - Ein Zeitraum von 2^{32} Sekunden (ca. 136 Jahre) mit einer Auflösung von 2^{-32} Sekunden (ca. 0,23 Nanosekunden) ist so darstellbar

Hierarchische Struktur eines Verbundes von NTP-Servern

- NTP nutzt ein hierarchisches System sogenannter *Strata*
  - Stratum 0 ist eine Atomuhr oder Funkuhr auf Basis des Zeitsignalsenders DCF77 oder des globalen Navigationssatellitensystems (GPS)
  - Stratum 1 sind die direkt mit Stratum 0 gekoppelten NTP-Server (*Zeitserver*)
  - Darunter folgen weitere Ebenen und die Endgeräte
  - Die Stratum-Ebene gibt den Abstand von Stratum 0 an



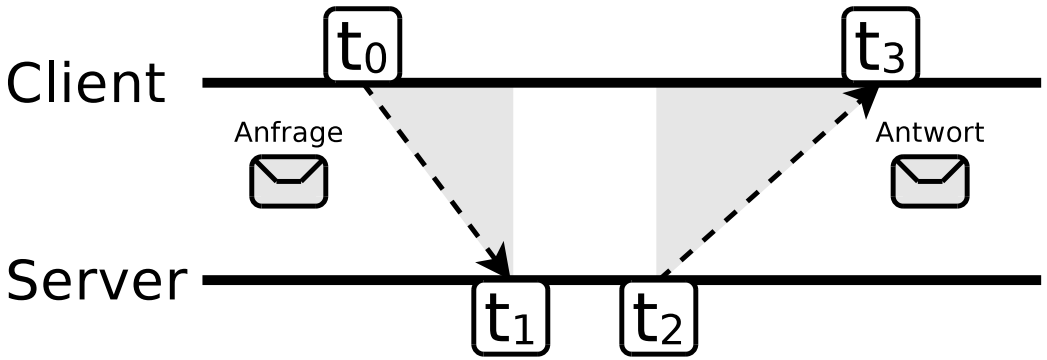
- Die NTP-Software auf Stratum 1, 2, usw. ist zugleich Client des darüber liegenden Stratums als auch Server für das darunter liegende Stratum, wenn es denn existiert
- NTP verwendet die UTC-Zeitskala
- $>$ 100.000 NTP-Knoten existierten weltweit

Eine NTP-Zeitquelle (Stratum 0)

U.S. Naval Observatory – Schriever Air Force Base in Colorado. Lizenz: CC0
Bildquelle: `http://www.af.mil/shared/media/photodb/photos/060104-F-3966R-005.jpg`

Zeit-Synchronisations-Algorithmus von NTP

- Um die lokale Uhr mit einem NTP-Server zu synchronisieren, muss ein NTP-Client, die Umlaufzeitverzögerung und die Abweichung berechnen
  - Zeitpunkt t_0: Client sendet Anfrage
  - Zeitpunkt t_1: Server empfängt Anfrage
  - Zeitpunkt t_2: Server sendet Antwort
  - Zeitpunkt t_3: Client empfängt Antwort
  - $t_3 - t_0 \Longrightarrow$ Zeitraum zwischen Senden und Empfangen des Clients
  - $t_2 - t_1 \Longrightarrow$ Zeitraum zwischen Empfangen und Senden des Servers



- Umlaufzeitverzögerung (Round Trip Delay Time) $= (t_3 - t_0) - (t_2 - t_1)$
- Abweichung (Offset) $= \frac{(t_1 - t_0) + (t_2 - t_3)}{2}$

Ausgabe des NTP-Daemons

- Meist fragt ein Client ≥ 3 NTP-Server in verschiedenen Netzen ab
  - Ausreißer werden verworfen
  - Eine geschätzte Abweichung (Offset) wird aus den besten Kandidaten berechnet

```
$ ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
+foxtrot.zq1.de  235.106.237.243  3 u  247 1024  277    49.765   -2.701  46.993
*ns2.customer-re 40.33.41.76      2 u  331 1024  377    50.853    0.390 234.340
+nono.com        78.46.60.42      3 u  746 1024  377    50.469    0.307  28.140
+thw23.de        52.239.121.49    3 u  969 1024  377    51.589    0.308  58.305
```

- Spalte 1: DNS-Name des NTP-Servers
- Spalte 2: IP des NTP-Servers
- Spalte 3: Stratum des NTP-Servers
- Spalte 4: Typ des NTP-Servers (`u` = Unicast)
- Spalte 5: Vergangene Sekunden seit der letzten Anfrage
- Spalte 6: Anfrageintervall in Sekunden
- Spalte 7: Wie häufig wurde der NTP-Server erfolgreich erreicht (377 = die letzten 8 mal)
- Spalte 8: `delay` = Round Trip Time
- Spalte 9: `offset` der lokalen Uhr gegenüber dem NTP-Server
- Spalte 10: `jitter` = Genauigkeitsschwankungen im Übertragungstakt

Telnet (Telecommunication Network)

- Protokoll (RFC 854) zur Fernsteuerung von Rechnern
  - Ermöglicht zeichenorientierten Datenaustausch über **TCP** via Port 23
  - Eignet sich nur für Anwendungen ohne grafische Benutzeroberfläche
- Software, die das Protokoll implementiert, heißt auch einfach Telnet
  - Besteht aus Telnet-Client und Telnet-Server
- Nachteil: **Keine Verschlüsselung!**
  - Auch die Passwörter werden im Klartext versendet
    $\Longrightarrow$ zu unsicher für entferntes Arbeiten
  - Nachfolger: Secure Shell (SSH)
- Wird häufig zur Fehlersuche bei anderen Diensten, zum Beispiel Web-Servern, FTP-Servern oder SMTP-Servern, und zur Administration von Datenbanken sowie in LANs eingesetzt
- Telnet-Clients können sich **mit beliebigen Portnummern verbinden**
  - Das ermöglicht dem Administrator, über einen Telnet-Client, Kommandos an Web-Server, FTP-Server oder SMTP-Server zu senden und unverfälscht deren Reaktion zu beobachten

Telnet und das virtuelle Netzwerkterminal

- Telnet basiert auf dem Standard NVT
  - NVT (Network Virtual Terminal) = virtuelles Netzwerkterminal
    - Telnet-Clients konvertieren die Tasteneingaben und Kontrollanweisungen in das NVT-Format und übertragen diese Daten an den Telnet-Server, der sie wiederum dekodiert und weiterreicht
  - NVT arbeitet mit Informationseinheiten von je 8 Bits (1 Byte)
    - NVT verwendet die 7-Bit-Zeichenkodierung US-ASCII
    - Das höchstwertige Bit jedes Zeichens wird mit Null aufgefüllt, um auf 8 Bits zu kommen

Name	Code	Beschreibung
NULL	NUL	No operation
Line Feed	LF	Zeilenvorschub (nächste Zeile, gleiche Spalte)
Carriage Return	CR	Wagenrücklauf (gleiche Zeile, erste Spalte)
BELL	BEL	Hörbares oder sichtbares Signal
Back Space	BS	Cursor eine Position zurück bewegen
Horizontal Tab	HT	Horizontaler Tabulatorstopp
Vertical Tab	VT	Vertikaler Tabulatorstopp
Form Feed	FF	Cursor in die erste Spalte der ersten Zeile bewegen und Terminal löschen

Die Tabelle enthält die Kontrollanweisungen von NVT

Die ersten 3 Kontrollzeichen versteht jeder Telnet-Client und -Server. Die übrigen 5 Kontrollzeichen sind optional

Secure Shell (SSH)

- Ermöglicht eine verschlüsselte und damit sichere Verbindung zwischen 2 Rechnern über ein unsicheres Netzwerk
  - Sichere Alternative zu Telnet
- Verwendet TCP und standardmäßig Port 22

- SSH-1 wurde 1995 von Tatu Ylönen entwickelt und als Freeware veröffentlicht
- Quelloffene Alternative: OpenSSH (`http://openssh.com`)
- SSH-2 wurde 1996 veröffentlicht und hat u.a. eine verbesserte Integritätsprüfung

- Beliebige TCP/IP-Verbindungen können über SSH getunnelt werden (Port-Weiterleitung)
  - Häufige Anwendung: X11-Anwendungen via SSH tunneln
- SSH-2 verwendet den Verschlüsselungsalgorithmus AES mit 128 Bit Schlüssellänge
  - Zudem werden 3DES, Blowfish, Twofish, CAST, IDEA, Arcfour, SEED und AES mit anderen Schlüssellängen unterstützt

Hypertext-Übertragungsprotokoll (HTTP)

- Das Hypertext Transfer Protocol (HTTP) ist ein zustandsloses Protokoll zur Übertragung von Daten
  - Zustandslos heißt, dass jede HTTP-Nachricht alle nötigen Informationen enthält, um die Nachricht zu verstehen
  - Der Server hält keine Zustands- bzw. Sitzungsinformation über den Client vor, und jede Anfrage ist eine von anderen Anfragen unabhängige Transaktion

HTTP

Ab 1989 von Roy Fielding, Tim Berners-Lee und anderen am CERN entwickelt

- Ist gemeinsam mit den Konzepten URL und HTML die Grundlage des World Wide Web (WWW)
- Haupteinsatzzweck: Webseiten aus dem World Wide Web (WWW) in einen Browser laden
- Zur Kommunikation ist HTTP auf ein zuverlässiges Transportprotokoll angewiesen
  - In den allermeisten Fällen wird TCP verwendet
- Jede HTTP-Nachricht besteht aus:
  - Nachrichtenkopf (*HTTP-Header*): Enthält u.a. Informationen zu Kodierung, gewünschter Sprache, Browser und Inhaltstyp
  - Nachrichtenkörper (*Body*): Enthält die Nutzdaten, wie den HTML-Quelltext einer Webseite

HTTP-Anfragen (1/2)

- Wird via HTTP auf eine URL (z.B. `http://www.informatik.hs-mannheim.de/~baun/index.html` zugegriffen, wird an den Rechner mit dem Hostnamen `www.informatik.hs-mannheim.de` eine Anfrage für die Ressource `/~baun/index.html` gesendet
- Zuerst wird der Hostname via DNS in eine IP-Adresse umgewandelt
- Über TCP wird zu Port 80, auf dem der Web-Server üblicherweise arbeitet, folgende HTTP-`GET`-Anforderung gesendet

```
GET /~baun/index.html HTTP/1.1
Host: www.informatik.hs-mannheim.de
User-Agent: Mozilla/5.0 (X11; U; Linux i686; de; rv:1.9.2.18) Gecko/20110628 Ubuntu/10.10 (
     maverick) Firefox/3.6.18
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: de-de,de;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
...
```

HTTP-Anfragen (2/2)

- So ein großer Nachrichtenkopf ist eigentlich nicht nötig
- Die hier angegebene HTTP-`GET`-Anforderung genügt völlig

```
GET /~baun/index.html HTTP/1.1
Host: www.informatik.hs-mannheim.de
```

- Der Nachrichtenkopf einer HTTP-Nachricht wird mit einem Line Feed (`LF`) und einem Carriage Return (`CR`) vom Nachrichtenkörper abgegrenzt
  - Im Beispiel hat die HTTP-Anforderung aber keinen Nachrichtenkörper

HTTP-Antworten (1/2)

- Die HTTP-Antwort des Web-Servers besteht aus einem Nachrichtenkopf und dem Nachrichtenkörper mit der eigentlichen Nachricht
  - In diesem Fall enthält der Nachrichtenkörper den Inhalt der angeforderten Datei `index.html`

```
HTTP/1.1 200 OK
Date: Sun, 04 Sep 2011 15:19:13 GMT
Server: Apache/2.2.17 (Fedora)
Last-Modified: Mon, 22 Aug 2011 12:37:04 GMT
ETag: "101ec1-2157-4ab17561a3c00"
Accept-Ranges: bytes
Content-Length: 8535
Keep-Alive: timeout=13, max=499
Connection: Keep-Alive
Content-Type: text/html

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
        "http://www.w3.org/TR/html4/loose.dtd">
<html>
...
</html>
```

HTTP-Antworten (2/2)

- Jede HTTP-Antwort enthält einen **Statuscode**, der aus 3 Ziffern besteht, und eine Textkette, die den Grund für die Antwort beschreibt

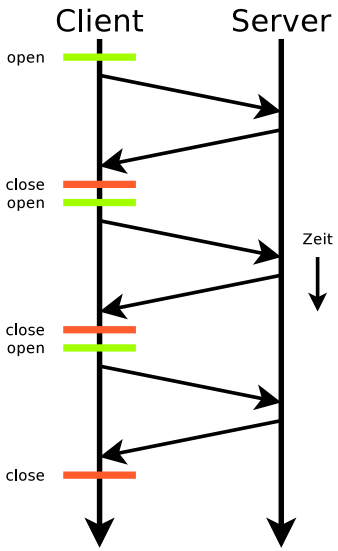
Statuscode	Bedeutung	Beschreibung
`1xx`	**Informationen**	Anfrage erhalten, Prozess wird fortgeführt
`2xx`	**Erfolgreiche Operation**	Aktion erfolgreich empfangen
`3xx`	**Umleitung**	Weitere Aktion des Clients erforderlich
`4xx`	**Client-Fehler**	Anfrage des Clients fehlerhaft
`5xx`	**Server-Fehler**	Fehler, dessen Ursache beim Server liegt

- Die Tabelle enthält einige bekannte Statuscodes von HTTP

Statuscode	Bedeutung	Beschreibung
200	`OK`	Anfrage erfolgreich bearbeitet. Ergebnis wird in der Antwort übertragen
202	`Accepted`	Anfrage akzeptiert, wird aber zu einem späteren Zeitpunkt ausgeführt
204	`No Content`	Anfrage erfolgreich durchgeführt. Antwort enthält bewusst keine Daten
301	`Moved Permanently`	Ressource verschoben. Die alte Adresse ist nicht länger gültig
307	`Temporary Redirect`	Ressource verschoben. Die alte Adresse bleibt gültig
400	`Bad Request`	Anfrage war fehlerhaft aufgebaut
401	`Unauthorized`	Anfrage kann nicht ohne gültige Authentifizierung durchgeführt werden
403	`Forbidden`	Anfrage mangels Berechtigung des Clients nicht durchgeführt
404	`Not Found`	Ressource vom Server nicht gefunden
500	`Internal Server Error`	Unerwarteter Serverfehler

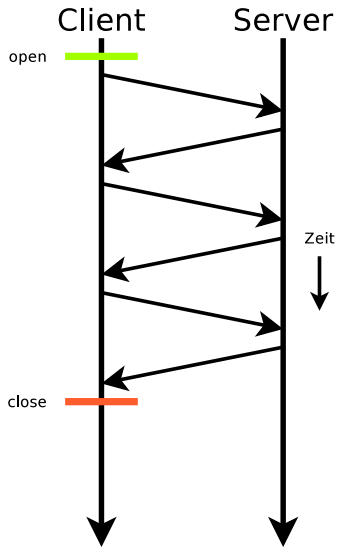
HTTP-Protokollversionen (HTTP/1.0 und HTTP/1.1)

- 3 Protokollversionen existieren: `HTTP/1.0`, `HTTP/1.1` und `HTTP/2`



- HTTP/1.0 (RFC 1945): Vor jeder Anfrage wird eine neue TCP-Verbindung aufgebaut und nach der Übertragung der Antwort standardmäßig vom Server wieder geschlossen
  - Enthält ein HTML-Dokument Referenzen auf zum Beispiel 10 Bilder, sind also 11 TCP-Verbindungen zur Übertragung an den Client nötig

- HTTP/1.1 (RFC 2616): Es wird standardmäßig kein Verbindungsabbau durchgeführt
  - Für den Transfer eines HTML-Dokuments mit 10 Bildern ist somit nur eine einzige TCP-Verbindung nötig
    - Dadurch wird das Dokument schneller geladen
  - Zudem können abgebrochene Übertragungen bei `HTTP/1.1` fortgesetzt werden



HTTP-Protokollversionen (HTTP/2)

- `HTTP/2` (RFC 7540) wurde im Mai 2015 standardisiert
- Beschleunigt die Datenübertragung u.a. durch eine Kompression des Headers mit dem Algorithmus HPACK (RFC 7541)
- Ermöglicht das Zusammenfassen (*Multiplex*) von Anfragen und ein Server kann von sich aus Daten senden (*Server Push*), von denen er weiß, dass sie der Browser umgehend benötigen wird
  - Beispiele für solche Daten sind CSS-Dateien (Cascading Style Sheets), die die Darstellung der Webseiten definieren, oder Script-Dateien
- `HTTP/2` ist kein textbasiertes, sondern ein binäres Protokoll
  - Darum kann nicht mit einfachen Werkzeugen wie `telnet` und `nc` darüber kommuniziert werden, um z.B. einen Server zu untersuchen
  - Werkzeuge wie `curl` und `openssl -connect` können via HTTP/2 kommunizieren

Einige Quellen zu `curl` und `openssl`

`https://stackoverflow.com/questions/51278076/curl-one-liner-to-test-http-2-support`
`https://blog.cloudflare.com/tools-for-debugging-testing-and-using-http-2/`
Stephen Ludin, Javier Garza. **Learning HTTP/2: A Practical Guide for Beginners**. O'Reilly Media, Inc (2017)

HTTP-Methoden

- Das HTTP-Protokoll enthält einige Methoden für Anfragen

HTTP	Beschreibung
`PUT`	Neue Ressource auf den Web-Server hochladen
`GET`	Ressource vom Web-Server anfordern
`POST`	Daten zum Web-Server hochladen, um Ressourcen zu erzeugen
`DELETE`	Eine Ressource auf dem Web-Server löschen
`HEAD`	Header einer Ressource vom Web-Server anfordern, aber nicht den Body
`TRACE`	Liefert die Anfrage so zurück, wie der Web-Server sie empfangen hat. Hilfreich für die Fehlersuche
`OPTIONS`	Liste der vom Web-Server unterstützten HTTP-Methoden anfordern
`CONNECT`	SSL-Tunnel mit einem Proxy herstellen

HTTP ist ein zustandsloses Protokoll. Über Cookies in den Header-Informationen sind dennoch Anwendungen realisierbar, die Status- bzw. Sitzungseigenschaften erfordern weil sie Benutzerinformationen oder Warenkörbe den Clients zuordnen.

Eine Möglichkeit, Web-Server zu testen, ist telnet (1/2)

```
$ telnet www.informatik.hs-mannheim.de 80
Trying 141.19.145.2...
Connected to anja.ki.fh-mannheim.de.
Escape character is '^]'.
GET /~baun/index.html HTTP/1.0

HTTP/1.1 200 OK
Date: Sun, 04 Sep 2011 21:43:53 GMT
Server: Apache/2.2.17 (Fedora)
Last-Modified: Mon, 22 Aug 2011 12:37:04 GMT
ETag: "101ec1-2157-4ab17561a3c00"
Accept-Ranges: bytes
Content-Length: 8535
Connection: close
Content-Type: text/html
X-Pad: avoid browser bug

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
        "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
...
</body>
</html>
Connection closed by foreign host.
```

Bei Verschlüsselung (HTTPS): `openssl s_client -connect <server>:<port>`

Eine Möglichkeit, Web-Server zu testen, ist `telnet` (2/2)

```
$ telnet www.informatik.hs-mannheim.de 80
Trying 141.19.145.2...
Connected to anja.ki.fh-mannheim.de.
Escape character is '^]'.
GET /~baun/test.html HTTP/1.0

HTTP/1.1 404 Not Found
Date: Sun, 04 Sep 2011 21:47:26 GMT
Server: Apache/2.2.17 (Fedora)
Content-Length: 301
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /~baun/test.html was not found on this server.</p>
<hr>
<address>Apache/2.2.17 (Fedora) Server at anja.ki.hs-mannheim.de Port 80</address>
</body></html>
Connection closed by foreign host.
```

Simple Mail Transfer Protocol (SMTP)

- Protokoll (RFC 5321) für den Austausch (Versand) von Emails
- Verwendet TCP und standardmäßig Port 25
- Das Abholen von Emails erfolgt mit den Protokollen POP3 oder IMAP
- Zum Versand von Emails verbindet sich das Mailprogramm des Benutzers mit einem SMTP-Server, der die Emails über ggf. weitere SMTP-Server zum Ziel weiterversendet
- Da SMTP ein textbasiertes Protokoll ist, kann man sich auch via Telnet mit einem SMTP-Server verbinden und so auch Emails *von Hand* versenden
  - Die Absender- und Empfängeradresse sind bei SMTP frei wählbar
    - Die Adressen im `MAIL FROM`- und `RCPT TO`-Kommando können sich von den Adressen in den Feldern `From` und `To` im Header der Email unterscheiden
  - Eine Authentifizierung findet nicht zwingend statt
    - In SMTP gibt also keine Verlässlichkeit der Absenderangabe in Emails

Statuscodes von SMTP-Servern

- Ein SMTP-Server antwortet auf Anfragen mit dreistelligen Statuscodes und kurzen Texten, die variieren oder entfallen können

Statuscode	Bedeutung	Beschreibung
2xx	**Erfolgreiche Ausführung**	Kommando erfolgreich ausgeführt
4xx	**Temporärer Fehler**	Wird das Kommando wiederholt, ist die Ausführung eventuell möglich
5xx	**Fataler Fehler**	Kommando kann nicht ausgeführt werden

- Die folgende Tabelle enthält einige SMTP-Kommandos

Kommando	Funktion
`HELO`	SMTP-Sitzung starten und Client identifizieren
`MAIL From:<...>`	Email-Adresse des Absenders angeben
`RCPT To:<...>`	Email-Adresse des Empfängers angeben
`DATA`	Inhalt der Email angeben
`RSET`	Eingabe einer Email abbrechen
`NOOP`	Keine Operation. Hält die Verbindung aufrecht
`QUIT`	Beim SMTP-Server abmelden

- Der Betrieb eines SMTP-Servers ist nicht ohne Sicherheitsrisiken
  - Mit Zusatzsoftware können SMTP-Server aber abgesichert werden (z.B. S/MIME für Signaturen, SSL/TLS für Verschlüsselung)

Populäre SMTP-Server: Exim, IBM Lotus Domino, MS Exchange, Postfix, Sendmail,...

Email via SMTP mit Telnet versenden

```
$ telnet sushi.unix-ag.uni-kl.de 25
Trying 2001:638:208:ef34:0:ff:fe00:65...
Connected to sushi.unix-ag.uni-kl.de.
Escape character is '^]'.
220 sushi.unix-ag.uni-kl.de ESMTP Sendmail 8.14.3/8.14.3/Debian-5+lenny1; Mon, 5 Sep...
HELO sushi
250 sushi.unix-ag.uni-kl.de Hello sushi.unix-ag.uni-kl.de, pleased to meet you
MAIL FROM:<cray@unix-ag.uni-kl.de>
250 2.1.0 <cray@unix-ag.uni-kl.de>... Sender ok
RCPT TO:<wolkenrechnen@gmail.com>
250 2.1.5 <wolkenrechnen@gmail.com>... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
From: <cray@unix-ag.uni-kl.de>
To: <wolkenrechnen@gmail.com>
Subject: Testmail
Date: Mon, 5 Sep 2011 11:49:50 +200

Das ist eine Testmail.
.
250 2.0.0 p859lbSc018528 Message accepted for delivery
QUIT
221 2.0.0 sushi.unix-ag.uni-kl.de closing connection
Connection closed by foreign host.
```

Bei Verschlüsselung (TLS): `openssl s_client -starttls smtp -connect <server>:587`
Bei Verschlüsselung (SSL): `openssl s_client -connect <server>:465`

Post Office Protocol (POP)

- Protokoll (RFC 918), das das Auflisten, Abholen und Löschen von Emails von einem Email-Server ermöglicht
- Verwendet TCP und standardmäßig Port 110
- Die aktuelle Version ist Version 3 (POP3) von 1988 (RFC 1081 und 1939)
- Die vollständige Kommunikation wird im Klartext übertragen
- Da POP3 ein textbasiertes Protokoll ist, kann man via Telnet Emails auch *von Hand* auflisten, abholen und löschen

Emails via Telnet auflisten, abholen und löschen (1/2)

Kommando	Funktion
`USER xxx`	Benutzernamen auf dem Server angeben
`PASS xxx`	Passwort angeben
`STAT`	Anzahl aller Emails im Postfach und deren Gesamtgröße (in Byte) ausgeben
`LIST (n)`	Nachrichtennummer(n) und Größe der (n-ten) Email(s) ausgeben
`RETR n`	Die n-te Email vom Server ausgeben
`DELE n`	Die n-te Email vom Server löschen
`RSET`	Alle `DELE`-Kommandos zurücksetzen
`NOOP`	Keine Operation. Hält die Verbindung aufrecht
`QUIT`	Am Server abmelden und die `DELE`-Kommandos ausführen

```
$ telnet pop.gmx.com 110
Trying 212.227.17.187...
Connected to pop.gmx.com.
Escape character is '^]'.
+OK POP server ready H migmx001
USER christianbaun@gmx.de
+OK password required for user "christianbaun@gmx.de"
PASS xyz
+OK mailbox "christianbaun@gmx.de" has 2 messages (6111 octets) H migmx107
STAT
+OK 2 6111
LIST
+OK
1 4654
2 1457
```

Emails via Telnet auflisten, abholen und löschen (2/2)

```
RETR 2
+OK
Return-Path: <wolkenrechnen@gmail.com>
Delivered-To: GMX delivery to christianbaun@gmx.de
...
From: Christian Baun <wolkenrechnen@gmail.com>
To: christianbaun@gmx.de
Subject: Testmail
Date: Mon, 5 Sep 2011 15:33:39 +0200
User-Agent: KMail/1.13.5 (Linux/2.6.35-30-generic; KDE/4.5.5; i686; ; )
MIME-Version: 1.0
Content-Type: Text/Plain;
  charset="us-ascii"
Content-Transfer-Encoding: 7bit
...

Das ist eine Testmail.
.
DELE 2
+OK
QUIT
+OK POP server signing off
Connection closed by foreign host.
```

Bei Verschlüsselung (POP3S): `openssl s_client -connect <server>:995`