10th Slide Set
Computer Networks

Prof. Dr. Christian Baun

Frankfurt University of Applied Sciences
(1971–2014: Fachhochschule Frankfurt am Main)
Faculty of Computer Science and Engineering
christianbaun@fb2.fra-uas.de

Learning Objectives of this Slide Set

- Session Layer
- Presentation Layer
- Application Layer
    - Application Layer protocols
        - Domain name resolution (DNS)
        - Automatic assignment of addresses (DHCP)
        - Time synchronization (NTP)
        - Remote control of computers (Telnet, SSH)
        - Transferring data (HTTP)
        - Exchange emails (SMTP)
        - Retrieve emails (POP3)
        - Upload and download files (FTP)

Session Layer

- Provides the functionality for establishment, monitoring and termination of sessions
    - A session is the basis for a virtual connection between two applications on physically independent computers
    - A session consists of requests and replies between applications
- Furthermore, this layer provides the dialogue control functionality (which participant speaks right now)
- Functions for synchronization
    - Checkpoints can be inserted into large data transmissions for session restoration
    - If the connection fails, the transmission can continue at the latest checkpoint and does not need to start again from the beginning
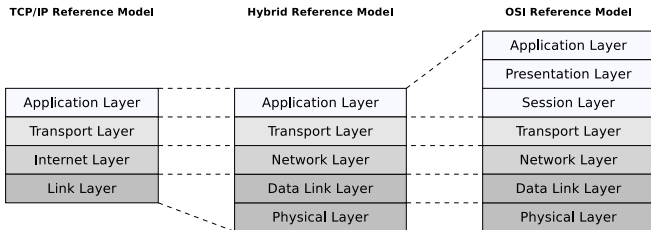
Session Layer Protocols

- Protocols, which meet the required features of the Session Layer, are e.g. Telnet for the remote controlling of computers and FTP for transferring files
    - However, these protocols can also be assigned to the Application Layer
- The Application Layer contains the protocols, which are used by the applications
- Telnet and FTP are used directly by the corresponding application programs, and not by abstract protocols at upper protocols layers
    - Therefore, it is useful to assign the Session Layer protocols to the Application Layer

Presentation Layer

- Contains rules for the formatting (presentation) of message
    - The sender can inform the receiver that a message is encoded in a specific format (e.g. ASCII)
        - Objective: Enable the receiver to do the necessary conversion
    - In this layer, data records can assigned to fields (e.g. name, student ID number. . .)
    - The type and length of data types can be specified here
    - Compression and encryption are assigned to the Presentation Layer
- Exactly like the Session Layer, the Presentation Layer is hardly used in practice
    - Reason: All features of this layer are provided by Application Layer protocols today

Application Layer

- Contains the protocols, which interact with applications (e.g. browser or email client)
- Contains the messages of the users and their applications (e.g. HTML pages or emails) in accordance with the Application Layer protocol used

TCP/IP Reference Model	Hybrid Reference Model	OSI Reference Model
		Application Layer
		Presentation Layer
Application Layer	Application Layer	Session Layer
Transport Layer	Transport Layer	Transport Layer
Internet Layer	Network Layer	Network Layer
Link Layer	Data Link Layer	Data Link Layer
	Physical Layer	Physical Layer

- Devices: none
- Protocols: DNS, DHCP, NTP, Telnet, SSH, HTTP, SMTP, FTP. . .

Domain Name System (DNS)

- Protocol for the **resolution** of domain names to IP addresses

RFC 1034 and 1035

- Similar to a telephone assistance
    - Person/family/company $\Longrightarrow$ telephone number
    - Hostname/website $\Longrightarrow$ IP address

Developed in 1983 by Paul Mockapetris

- DNS replaced the local domain name tables in the config file
  /etc/hosts, which until then had been used for managing the domain
  names/IP addresses mappings
    - These were no longer able to handle the growing number of new entries
- Bases on a hierarchical namespace
    - The assignment records are split into separate parts and distributed to
      name servers across the internet

Domain Namespace (1/2)

- The domain namespace consists of a **tree of domain names**
    - Leafs and nodes are called **labels**
    - Each subtree is a **domain**
- A complete domain name consists of the concatenation of all labels of a path
- Labels are alphanumeric character strings
    - The dash (-) is the only special character allowed
    - The length of a label must be at least 1 and can be up to 63 characters
    - Labels must start with an letter and can not start or end with a dash
    - Each labels ends with a dot
- Domain names end with a dot
    - The dot is usually omitted, but from a formal perspective, a complete domain name – **Fully Qualified Domain-Name** (FQDN) ends with a dot
- An example for a complete domain name is `www.h-da.de.`

Domain Namespace (2/2)
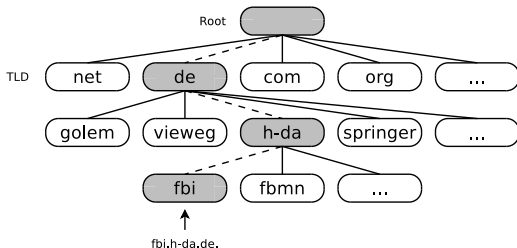
- Domain names are resolved from right to left
  - The further right a label is, the upper located is it in the tree

- The first layer below root is called **top level domain** (TLD)
- The DNS objects of a domain (e.g. the hostname) are stored as a set of **resource records** (RR) in a zone file, which is stored at one or more name servers
- The zone file is often simply called **zone**

Root Name Server (1/2)

- The 13 root servers (A to M) publish the DNS **root zone**
    - Their domain names have the form letter.root-servers.net
    - The root zone contains approx. 3000 entries and is the root of the DNS
        - It contains the hostnames and IP addresses of the name servers, which are responsible for the TLDs
- Some root servers do not consist of a single, but multiple physical servers, which are connected to a logical server
    - These computers are located at different locations around the world and they can be reached via **anycast** using the same IP address

Name	IPv4 address	IPv6 address	Location
A	198.41.0.4	2001:503:ba3e::2:30	distributed (anycast)
B	192.228.79.201	—	California, USA
C	192.33.4.12	—	distributed (anycast)
D	128.8.10.90	2001:500:2d::d	Maryland, USA
E	192.203.230.10	—	California, USA
F	192.5.5.241	2001:500:2f::f	distributed (anycast)
G	192.112.36.4	—	distributed (anycast)
H	128.63.2.53	2001:500:1::803f:235	Maryland, USA
I	192.36.148.17	2001:7fe::53	distributed (anycast)
J	192.58.128.30	2001:503:c27::2:30	distributed (anycast)
K	193.0.14.129	2001:7fd::1	distributed (anycast)
L	199.7.83.42	2001:500:3::42	distributed (anycast)
M	202.12.27.33	2001:dc3::35	distributed (anycast)

Root Name Server (2/2)

Image source: Wikipedia

- Before anycast was used, 10 root name servers were located in the US

- This was criticized because the geographical centralization was contrary to the decentralization paradigm of the internet
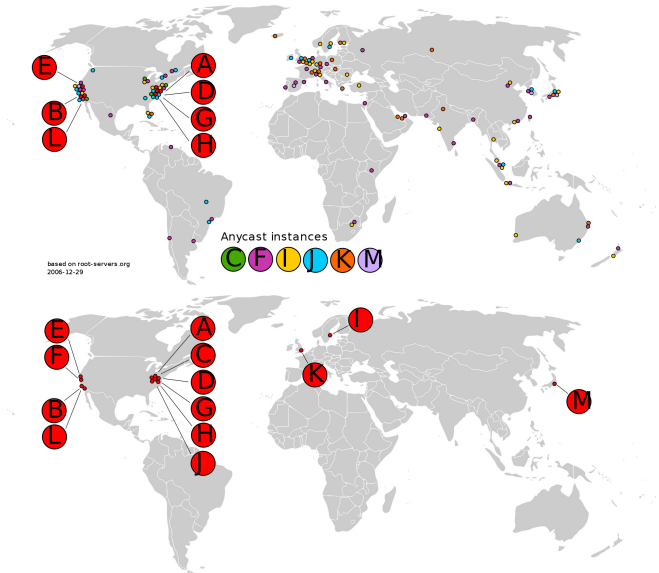
Structure of the DNS Database and the Resource Records

You already know. . .

- DNS implements a distributed database with a tree structure
- The data of the DNS of the internet is stored on globally distributed servers, with are linked to each other via references (*delegations*)
- In each name server, ≥ 1 zone files exist

- The zone files contain lists of resource records (RR)
- Every RR consists of 5 elements
  `<Name, Value, Type, Class, TTL>`
- The table contains some types of RRs

Type	Description
NS	Specifies the name server which is responsible for the zone or links zones to a tree of zones (delegation)
A	Specifies the IPv4 address of a host
AAAA	Specifies the IPv6 address of a host
SOA	Contains information for the management of the zone, such as the name and email address of the administrator
CNAME	Specifies an alias domain name for a specific host
MX	Assigns a SMTP mail server to a name.
	All other services use CNAME, A and AAAA resource records for the name resolution
PTR	Assigns an IP address to one or more hostname(s).
	Counterpart to the usual assignment of one or more IPs to a host name via an A or AAAA resource record

Example of a Domain Name Resolution (1/5)

- In this example, the domain name www.fh-frankfurt.de. is resolved with the command line tool dig

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

- -t A ⟹ request the A resource record (the IPv4 address)
- +trace ⟹ print the individual replies on the path through the name server hierarchy
- +additional ⟹ name servers sometimes store for delegations not only the NS resource records, but also their IP addresses in form of A or AAAA RRs. Print them, if they are delivered

- To resolve this IP, 4 name servers must be consulted successively

Example of a Domain Name Resolution (2/5)

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

; <<>> DiG 9.7.1-P2 <<>> +trace +additional -t A www.fh-frankfurt.de.
;; global options: +cmd
.                       3600000 IN      NS      a.root-servers.net.
.                       3600000 IN      NS      l.root-servers.net.
.                       3600000 IN      NS      j.root-servers.net.
.                       3600000 IN      NS      b.root-servers.net.
.                       3600000 IN      NS      c.root-servers.net.
.                       3600000 IN      NS      f.root-servers.net.
.                       3600000 IN      NS      d.root-servers.net.
.                       3600000 IN      NS      e.root-servers.net.
.                       3600000 IN      NS      k.root-servers.net.
.                       3600000 IN      NS      h.root-servers.net.
.                       3600000 IN      NS      i.root-servers.net.
.                       3600000 IN      NS      g.root-servers.net.
.                       3600000 IN      NS      m.root-servers.net.
;; Received 241 bytes from 10.0.0.1#53(10.0.0.1) in 46 ms
```

- In the last row is 10.0.0.1 the IP address of the name server of the requesting host
    - This name server knows the IP addresses of the root name servers
    - The IP addresses of the root name servers change seldom and must be well-known by all name servers, if they answer requests concerning the internet

Example of a Domain Name Resolution (3/5)

```
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    n.de.net.
de.                   172800  IN    NS    s.de.net.
de.                   172800  IN    NS    z.nic.de.
a.nic.de.             172800  IN    A     194.0.0.53
f.nic.de.             172800  IN    A     81.91.164.5
l.de.net.             172800  IN    A     77.67.63.105
n.de.net.             172800  IN    A     194.146.107.6
s.de.net.             172800  IN    A     195.243.137.26
z.nic.de.             172800  IN    A     194.246.96.1
a.nic.de.             172800  IN    AAAA  2001:678:2::53
f.nic.de.             172800  IN    AAAA  2a02:568:0:2::53
l.de.net.             172800  IN    AAAA  2001:668:1f:11::105
n.de.net.             172800  IN    AAAA  2001:67c:1011:1::53
;; Received 351 bytes from 199.7.83.42#53(l.root-servers.net) in 79 ms
```

- From the 13 root name servers, l.root-servers.net was randomly chosen, to send it the request for www.fh-frankfurt.de.
- The reply contains 6 name servers (delegations) to choose from, which are responsible for the zone de.
    - For 4 servers, the request is also possible using IPv6 (AAAA)

Example of a Domain Name Resolution (4/5)

```
fh-frankfurt.de.              86400    IN     NS      deneb.dfn.de.
fh-frankfurt.de.              86400    IN     NS      medusa.fh-frankfurt.de.
fh-frankfurt.de.              86400    IN     NS      chaplin.rz.uni-frankfurt.de.
deneb.dfn.de.                 86400    IN     A       192.76.176.9
medusa.fh-frankfurt.de. 86400    IN     A       192.109.234.209
chaplin.rz.uni-frankfurt.de. 86400 IN     A       141.2.22.74
;; Received 169 bytes from 77.67.63.105#53(l.de.net) in 54 ms
```

- From the 6 name servers in the reply, l.de.net has been randomly chosen, to send it the request for www.fh-frankfurt.de.
- The reply contains 3 name servers (delegations) to choose from, which are responsible for the zone fh-frankfurt.

Example of a Domain Name Resolution (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      medusa.fh-frankfurt.de.
fh-frankfurt.de.          86400   IN      NS      deneb.dfn.de.
medusa.fh-frankfurt.de. 86400     IN      A       192.109.234.209
;; Received 139 bytes from 192.109.234.209#53(medusa.fh-frankfurt.de) in 57 ms
```

- From the 3 name servers in the reply, medusa.fh-frankfurt.de has been randomly chosen, to send it the request for www.fh-frankfurt.de.
- Result: 192.109.234.216

The DNS protocol

- DNS requests are usually sent via UDP port 53 to the server name
- The maximum length of a DNS reply via UDP is 512 bytes
- Longer DNS replies sends a Nameserver via TCP

Dynamic Host Configuration Protocol (DHCP)

- Allows the assignment of the network configuration (IP address, network mask, default gateway, name server, etc.) to network devices from a **DHCP server** by using a **DHCP client**
  - Especially for mobile devices, it is not useful to assign static IPs
    - Without DHCP, the network settings of all clients need to be customized after modifying the network topology
    - With DHCP, just the DHCP server's configuration need to be adjusted
- Uses UDP via ports 67 (server or relay agent) and 68 (client)

RFC 2131

- A DHCP server has a **pool of IPs** and distributes them to clients
- A DHCP client can only use a DHCP server, when both are connected to the **same logical network**
  - Reason: DHCP uses **broadcasts** and Routers do not forward broadcasts

If the DHCP server is connected with a different logical network, a **DHCP relay** need to forward the requests to the DHCP server

Functioning of DHCP (1/2)

1. A client without an IP address sends via **broadcast** a request (**DHCP Discover**) to the DHCP servers, which can be reached by it
   - Multiple DHCP servers might exist in a subnet
   - The sender IP address of the broadcast is 0.0.0.0
   - The destination address is 255.255.255.255
2. Each DHCP server, which receives the **broadcast** and has free IPs in its pool, responds to the request with an address offering (**DHCP Offer**)
   - The address offering is sent as **broadcast** too with destination address 255.255.255.255
3. The DHCP client accepts an address offering by sending a request (**DHCP-Request**) via broadcast
   - The message contains the ID of the desired DHCP server
   - The other (possibly existing) DHCP servers understand the message as a rejection of their address offers
4. The server responds with **DHCP Ack** and sets the IP address in its address pool as assigned
   - It can also refuse the request with **DHCP Nak**

Functioning of DHCP (2/2)
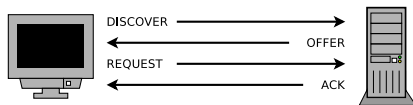
- If a DHCP server has assigned an IP and acknowledged this via **DHCP Ack**, it creates a *lease* record for the address in its database
    - If all addresses are assigned (leased), no more clients can be supplied with IP addresses
- Each address has an expiration date (*lease time*), which is transmitted to the client via the acknowledgement (**DHCP Ack**)
    - Active clients regularly renew the lease after the half lease time has expired via **DHCP Request**, which is sent via **unicast** directly to the server and not via broadcast
    - The server again responds with an acknowledgement (**DHCP Ack**), which contains the same data as before and a new expiration date
    - If the expiration date has expired, the server can assign the address new, when requests arrive

Structure of DHCP Messages

32 bits (4 bytes)

Operation	Type	Length	Hops
ID of the connection			
Seconds		Flags	
IP address (client)			
Own IP address			
IP address (server)			
IP address (relay)			
MAC address (client) (16 bytes)			
Name of server (64 bytes)			
Filename (128 bytes)			
DHCP parameters and options			

- **Operation** specifies the sort of the DHCP message
  - 1 = Request of a Client
  - 2 = Reply of a Server
- **Type** specifies the network technology
  - 1 = Ethernet, 6 = WLAN
- **Length** contains the length of the physical network address in bytes
- **Hops** is optional and contains the number of DHCP Relays on the path
- **Flags** indicates if the client still has a valid IP address
- **Filename** is optional and contains the name of a file, which the client is supposed to fetch via Trivial File Transfer Protocol (TFTP)
  - This allows a terminal device to boot via the network

Network Time Protocol (NTP)

- Standard for clock synchronization between computer systems

RFC 5905 describes the protocol and algorithms in detail

- NTP is the name of the protocol and of the reference implementation
  - Uses UDP via port 123

Developed in 1985 by David L. Mills of the University of Delaware

- The local clock is synchronized by the local background process (daemon) of the NTP software with an external time signal (e.g. atomic clock, local radio receiver or remote NTP servers via NTP)
- The timestamps in NTP have a length of 64 bits
  - 32 bits contain the *UNIX time* (seconds since 1.1.1970 00:00:00)
  - 32 bits contain the fractional second
  - Therefore, NTP can be used for a time scale of 2^{32} seconds (approx. 136 years) and it has a resolution of 2^{-32} seconds (0.23 nanoseconds)

Hierarchical Structure of a Network of NTP Servers

- NTP uses a hierarchical system of so-called *strata*
  - Stratum 0 is an atomic clock or a radio clock based on the time signal transmitter DCF77 or the Global Navigation Satellite System GPS
  - Stratum 1 are the NTP servers (*time servers*), which are coupled directly to stratum 0
  - Several lower levels exist, which contain among others the terminal devices
  - The stratum level specifies the distance from stratum 0

Stratum 0 (atomic or radio clock)

Stratum 1

Stratum 2

Stratum 3

Stratum 4 (user/terminal device)
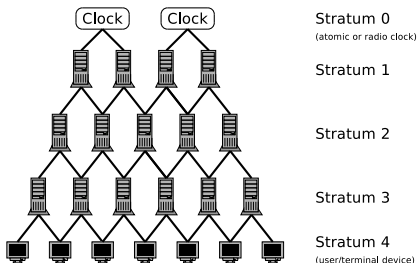
- The NTP software on stratum 1, 2 and so on, acts as client for the overlying stratum and as server for the underlying stratum, if it exists
- NTP uses the UTC time scale
- $> 100,000$ NTP nodes exist worldwide

A Stratum 0 Clock Source for NTP

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

Clock Synchronization Algorithm of NTP

- To synchronize its local clock with a remote NTP server, a NTP client needs to compute the round-trip delay time and the offset
    - Timestamp t_0: Client sends the request
    - Timestamp t_1: Server receives the request
    - Timestamp t_2: Server sends the reply
    - Timestamp t_3: Client receives the reply
    - $t_3 - t_0 \implies$ time elapsed on client side between the request is send and the reply is received
    - $t_2 - t_1 \implies$ time elapsed on server side between the request is received and the reply is send
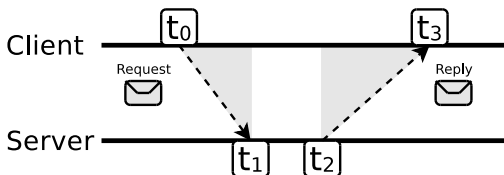
- Round trip delay time $= (t_3 - t_0) - (t_2 - t_1)$
- Offset $= \frac{(t_1 - t_0) + (t_2 - t_3)}{2}$

Output of the NTP Daemon

- Typically, a NTP client polls $\geq$ 3 NTP servers in different networks
    - Outliers are discarded
    - An estimate time offset is calculated from the best candidates

```
$ 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
```

- 1st column: DNS name of NTP server used
- 2nd column: IP of NTP server used
- 3rd column: Stratum of the NTP server
- 4th column: Type of NTP server (u = Unicast)
- 5th column: when was the last request (in seconds)
- 6th column: Interval of requests
- 7th column: How often the NTP server was successfully reached (377 = the last 8 times)
- 8th column: delay = Round Trip Time
- 9th column: offset of the local clock against the NTP server
- 10th column: jitter = deviation of the transmission timing

Telnet (Telecommunication Network)

- Protocol (RFC 854) for the remote control of computers
    - Provides text-oriented communication via **TCP** (default port: 23)
    - Suitable only for applications without a graphical user interface
- Software, which implements the protocol, is also simply called Telnet
    - Consists of the Telnet client and Telnet server
- Drawback: **No encryption!**
    - Also, the passwords are transmitted as plain text
      $\implies$ insufficient security for remote work
    - Successor: Secure Shell (SSH)
- Is often used for investigating issues of different services, such as web servers, FTP servers or SMTP servers, and for the administration of databases, and it is used in LANs
- Telnet clients are able to **connect to any port number**
    - This enables the administrator to send via a Telnet client requests (commands) to web servers, FTP servers or SMTP servers without an intermediate step and observe their reaction

Telnet and the Virtual Network Terminal

- Telnet is based on the NVT standard
    - NVT (Network Virtual Terminal) = virtual network terminal
        - Vendor independent interface
        - Concept for converting different data formats
        - Is implemented by all Telnet implementations on all hardware platforms
        - An NVT consists of an input device and an output device, which can generate and display only certain characters
        - Telnet clients convert the keystrokes and control statements in the NVT format and transmit this data to the Telnet server, which in turn decodes and forwards them
- NVT uses data units, each of a size of 8 bits (1 byte)
- NVT uses the 7-bit US-ASCII character encoding
- The most significant bit of each character is filled with a zero bit

Control Signals of Telnet

- The table contains the control signals of NVT
  - The first 3 control signals are implemented by each Telnet client and server
  - The remaining 5 control signals are optional

Name	Code	Description
NULL	NUL	No operation
Line Feed	LF	Moves the printer to the next line (keeps the horizontal position)
Carriage Return	CR	Moves the printer to the 1st row of the current line
BELL	BEL	Produces an audible or visible signal
Back Space	BS	Moves the printer one position back
Horizontal Tab	HT	Moves the printer to next horizontal tab stop
Vertical Tab	VT	Moves the printer to the next vertical tab stop
Form Feed	FF	Clears the terminal. Moves the printer to the 1st row of the 1st line

Secure Shell (SSH)

- Provides an encrypted and therefore secure communication between 2 hosts over an insecure network
  - Secure alternative to Telnet
- Uses TCP (default port: 22)

- SSH-1 was developed in 1995 by Tatu Ylönen and released as freeware
- Open Source alternative: OpenSSH (http://openssh.com)
- SSH-2 was released in 1996 and provides, among others, improved integrity checking

---

- Any TCP/IP connection can be tunneled over SSH (port forwarding)
  - Common application: Tunneling an X11 applications via SSH
- SSH-2 uses the AES encryption algorithm with a key length of 128 bits
  - 3DES, Blowfish, Twofish, CAST, IDEA, Arcfour, SEED and AES with other key lengths are supported too

Hypertext Transfer Protocol (HTTP)

- The Hypertext Transfer Protocol (HTTP) is a stateless protocol for data transmission
    - Stateless means that every HTTP message contains all the information necessary to understand the message
    - The server does not maintain any information regarding the state or session for the client, and each request is a transaction, independent of other requests

HTTP

From 1989 onwards, developed by Roy Fielding, Tim Berners-Lee and other at CERN

- Together with the concepts of URL and HTML it is the basis of the World Wide Web (WWW)
- Main purpose: Loading web pages from the World Wide Web (WWW) in a browser
- For communication, HTTP needs a reliable transport protocol
    - In almost all cases, TCP is used
- Each HTTP message consists of:
    - Message header (*HTTP header*): Includes among others Information about the encoding, desired language, browser and content type
    - Message body (*body*): Contains the payload, e.g. the HTML source code of a web page

HTTP Requests (1/2)

- If an URL is accessed via HTTP (e.g.
  http://www.informatik.hs-mannheim.de/~baun/index.html, the
  request for the resource /~baun/index.html is transmitted to the
  computer with hostname www.informatik.hs-mannheim.de

- First, via DNS, the hostname is resolved to an IP address

- Next, this HTTP GET request is transmitted via TCP to port 80, where
  the web server usually operates

```
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 Requests (2/2)

- A this large message header is not required
- The HTTP GET request below is sufficient

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

- The header of a HTTP message is separated from the message body with a line feed (LF) and a carriage return(CR)
  - In this example, the HTTP request has no message body

HTTP Responses (1/2)

- The HTTP response of the web server consists of a message header and the message body with the actual message
    - In this case, the message body contains the content of the requested file 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 Responses (2/2)

- Each HTTP response contains a **status code**, which consists of 3 digits, and a text string, which describes the reason for the answer

Status code	Meaning	Description
1xx	**Informational**	Request received, continuing process
2xx	**Success operation**	Action received, understood, accepted and processed successfully
3xx	**Redirection**	Additional action must be taken by the client to complete the request
4xx	**Client error**	Request of the client caused an error situation
5xx	**Server error**	Server failed to fulfill a valid request $\implies$ error was caused by server

- The table contains some common status codes of HTTP

Status code	Meaning	Description
200	OK	Request processed successfully. Result is transmitted in the response
202	Accepted	Request accepted, but will be executed at a later point in time
204	No Content	Request executed successfully. Response intentionally contains no data
301	Moved Permanently	The old address is no longer valid
307	Temporary Redirect	Resource moved. The old address remains valid
400	Bad Request	Request cannot be fulfilled due to bad syntax
401	Unauthorized	Request can not be executed without a valid authentication
403	Forbidden	Request is executed because of clients lack of privileges
404	Not Found	Server could not find the requested resource
500	Internal Server Error	Unexpected server error

HTTP/1.0 and HTTP/1.1
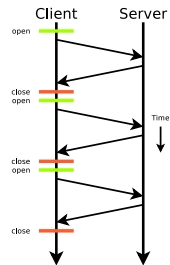
- 2 protocol versions exist: `HTTP/1.0` and `HTTP/1.1`
- HTTP/1.0 (RFC 1945): Prior to any request, a new TCP connection is established and closed per default by the server after the transmission of the response
    - If a HTML document contains e.g. 10 images, 11 TCP connections are required for the transmission to the client

- HTTP/1.1 (RFC 2616): Per default, no connection termination is done
    - So the connection can be used again and again
    - Therefore, only a single TCP connection is required for the transfer of a HTML document with 10 images
        - Result: The document download finishes in a shorter time
    - Interrupted transmissions can be resumed with `HTTP/1.1`
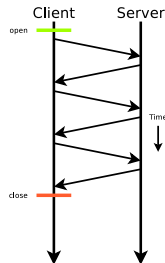
HTTP Methods

- The HTTP protocol provides some methods for requests

HTTP	Description
PUT	Upload a new resource to the web server
GET	Request a resource from the web server
POST	Upload data to the web server in order to generate resources
DELETE	Erase a resource on the web server
HEAD	Request the header of a resource from the web server, but not the body
TRACE	Returns the request back, as the web server has received it. Helpful for troubleshooting purposes
OPTIONS	Request the list of supported HTTP methods from the web server
CONNECT	Establish a SSL tunnel with a proxy

HTTP is a stateless protocol. But via cookies in the header information, applications can be implemented which require state or session information because they assign user information or shopping carts to clients.

Web Servers can be tested via `Telnet` (1/2)

```
$ telnet www.informatik.hs-mannheim.de 80
Trying 141.19.145.2...
Connected to anja.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.
```

When encryption is used: `openssl s_client -connect www.google.de:443 -crlf`

Web Servers can be tested via `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.
```

Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe

← → ▾ C ⊗ ⌂ ⬚ http://www.informatik.hs-mannheim.de/~baun/test.html ▾ Google.de

Not Found

The requested URL /~baun/test.html was not found on this server.

Apache/2.2.17 (Fedora) Server at www.informatik.hs-mannheim.de Port 80

Simple Mail Transfer Protocol (SMTP)

- Protocol (RFC 5321), which allows the exchange (delivery) of emails
- Uses TCP (default port: 25)
- Fetching emails is done with the protocols POP3 or IMAP
- For sending emails, the user's mail program connects to a SMTP server, which sends the emails, if necessary via additional SMTP servers, to the target
- Because SMTP is a text-based protocol, it is possible to connect via telnet to a SMTP server and send emails *manually*
    - The sender and destination addresses are freely selectable with SMTP
        - The addresses in the commands MAIL FROM and RCPT TO can be different from the addresses in the fields From and To in the header of the email
    - Authentication is not required and does not take place per default
        - In SMTP, the reliability of the sender information in emails is not implemented

Status Codes (Reply Codes) of SMTP Servers

- A SMTP server replies to a request with a three digit status/reply code and a short text which may vary or be omitted

Status code	Meaning	Description
2xx	**Success**	Command executed successfully
4xx	**Temporary failure**	Executing the command may be successful in the future
5xx	**Permanent failure**	Command can not be executed

- The table below contains some SMTP commands

Command	Function
HELO	Start SMTP session and identify client
MAIL From:<...>	Enter email address of the sender
RCPT To:<...>	Enter email address of the receiver
DATA	Enter Content of the email
RSET	Abort to enter an email
NOOP	No operation. Keeps the connection (avoids timeouts)
QUIT	Log out from the SMTP server

Sending Emails via SMTP with Telnet

```
$ 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

This is a test mail.
.
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.
```

Popular SMTP servers are among others Exim, IBM Lotus Domino, MS Exchange, Postfix and Sendmail

Security with SMTP

- Because of the lack of security features is the operation of a SMTP server risky
  - But additional software exists, which extends the functionality of SMTP servers

Feature	Meaning
Access control	Only registered users are allowed to use the server
	Potential solutions: SMTP-Auth, SMTPS
Authenticity	Explicit association of sender and message is possible
	Potential solutions: PGP, S/MIME (electronic signature)
Integrity	Message can not be altered without being noticed while being sent
	Potential solutions: PGP, S/MIME
Confidentiality	Message is transmitted encrypted
	Potential solutions: PGP, S/MIME, SSL/TLS

Post Office Protocol (POP)

- Protocol (RFC 918), which allows to list, fetch and delete emails from an email server
- Uses TCP (default port: 110)
- Latest version is version 3 (POP3) from 1988 (RFC 1081 and 1939)
- The entire communication is transmitted in plain text
- Because POP3 is a text-based protocol, it is possible to connect via telnet to a POP3 server and list, fetch and delete emails *manually*

List, Fetch and Delete Emails via Telnet (1/2)

Command	Function
USER xxx	Enter authentication username
PASS xxx	Enter password
STAT	Print the total number of emails in the mailbox and the total size (in bytes)
LIST (n)	Print the message numbers and size of all emails or of a specific email
RETR n	Print a specific email from the server
DELE n	Erase a specific email from the server
RSET	Reset all DELE commands
NOOP	No operation. Keeps the connection (avoids timeouts)
QUIT	Disconnect from the server and execute the DELE commands

```
$ 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
```

List, Fetch and Delete Emails via Telnet (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
...

This is a test mail.
.
DELE 2
+OK
QUIT
+OK POP server signing off
Connection closed by foreign host.
```

File Transfer Protocol (FTP)

- Protocol (RFC 959), which allows to download and upload files from/to FTP servers
- The entire communication via FTP is transmitted in plain text
- If FTP server and client are connected to each other, a TCP connection (**control port**) between exists between them
    - Via the connection, the control signals are sent to the server
    - FTP uses per default port 21 for the control port
    - For each command, the server responds with a status code (reply code) and in most cases with a a text, which describes the response
- For each operation, a separate TCP connection (**data port**) is established
    - These connections are used to send and receive files and for the transfer of directory listings
    - FTP uses per default port 20 for the data port
    - FTP connections can be established in **active mode** or **passive mode**

Active Mode and Passive Mode of FTP

- **Active mode**
  - The client opens a port with a port number > 1023 and tells the server via the command PORT what its IP and port number is
  - Next, the server establishes the FTP connection between its port 20 and the port of the client
    - **The connection establishment is initiated by the server**

- **Passive mode**
  - The client sends the command PASV to the server
  - The server opens a port and sends a message to the client, in which it tells the client the port number
  - The client opens a port with a port number > 1023 and establishes the FTP connection between its port and the server port
    - **The connection establishment is initiated by the client**
  - Passive mode is used e.g. when the client is located behind a firewall

Status Codes of FTP Servers and FTP Commands

- A FTP server responds to each command/request with a status code (reply code), which indicates the status of the communication

Status Code	Meaning
1xx	Command accepted, but execution is not yet finished
2xx	Command executed successfully
3xx	Client need to provide more information
4xx	Temporary failure. Executing the command may be successful in the future
5xx	Permanent failure. Command can not be executed

- The table below contains some FTP commands

Command	Function
ABOR	Abort an active file transfer
CWD	Change Working Directory
DELE	Erase file
LIST	Print information regarding a file or the current working directory
NOOP	No operation. Keeps the connection (avoids timeouts)
PASS	Enter password
PASV	Enter passive mode
PORT	Enter IP and port number for active mode
PWD	Print Working Directory
QUIT	Disconnect from the server
SIZE	Print the size of a file.
STAT	Print the current status
USER	Enter authentication username

Use a FTP servers via Telnet (1/2)

- Because FTP is a text-based protocol, it is possible to work via Telnet on a FTP server

```
$ telnet ftp.kernel.org 21
Trying 130.239.17.5...
Connected to pub.eu.kernel.org.
Escape character is '^]'.
220 Welcome to ftp.kernel.org.
USER anonymous
331 Please specify the password.
PASS guest
230-                       Welcome to the
230-
230-                    LINUX KERNEL ARCHIVES
230-                        ftp.kernel.org
230-
230-              "Much more than just kernels"
...
230 Login successful.
PASV
227 Entering Passive Mode (199,6,1,165,95,157).
PWD
257 "/"
NOOP
200 NOOP ok.
```

Use a FTP servers via Telnet (2/2)

```
STAT
211-FTP server status:
     Connected to 84.171.167.112
     Logged in as ftp
     TYPE: ASCII
     No session bandwidth limit
     Session timeout in seconds is 300
     Control connection is plain text
     Data connections will be plain text
     At session startup, client count was 56
     vsFTPd 2.3.4 - secure, fast, stable
211 End of status
CWD /pub
250 Directory successfully changed.
PWD
257 "/pub"
SIZE README
213 1912
SIZE index.html
213 2322
QUIT
221 Goodbye.
Connection closed by foreign host.
```

- An encrypted alternative to FTP is the Secure File Transfer Protocol (SFTP)