Computer Networks

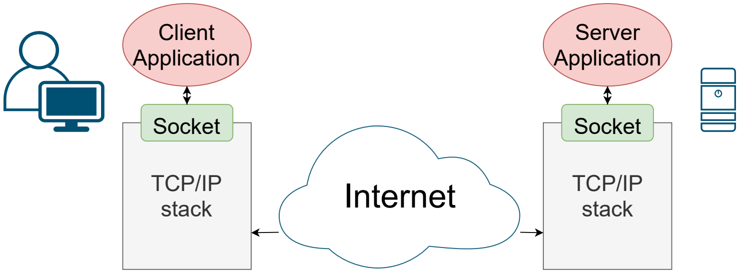
Application Layer

Prof. Dr. Oliver Hahm

Frankfurt University of Applied Sciences
Faculty 2: Computer Science and Engineering
`oliver.hahm@fb2.fra-uas.de`
`https://teaching.dahahm.de`

February 03, 2023

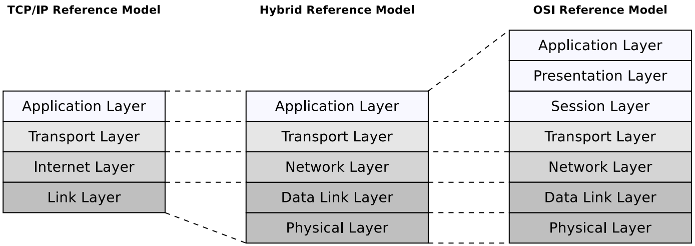
Networking Applications



- TCP/IP allows us to let two processes communicate over the Internet
- The socket interface is basically everything you need to develop a networking application
- But standardized protocols are helpful to be used on top of TCP/IP
- Some auxiliary protocols are almost essential, e.g., DHCP or DNS

Application Layer

- Contains the protocols, which interact with applications (e.g., web 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
- May be binary or human readable ($\rightarrow$ ASCII-encoded)



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

Agenda

- DNS
- Remote Shells
- HTTP
- E-Mail
- More Protocols

Agenda

- DNS
- Remote Shells
- HTTP
- E-Mail
- More Protocols

Domain Name System (DNS)

- Protocol for resolving (human-readable) domain names into (numeric) IP addresses
- Specified in RFC 1034 and 1035 and originally created by Paul Mockapetris
- Uses UDP via port 53
  - → UDP introduces less latency
  - → UDP requires no state
  - → DNS messages are small enough to fit into one UDP datagram
  - → DNS queries are idempotent ⇒ a timeout on the application layer is sufficient

Name Service for the Internet

- Similar to a telephone assistance
  - Person/family/company $\Longrightarrow$ telephone number
  - Domain name $\Longrightarrow$ IP address
- Bases on a hierarchical namespace
  - The assignment records are split into separate parts and distributed to name servers across the internet

/etc/hosts

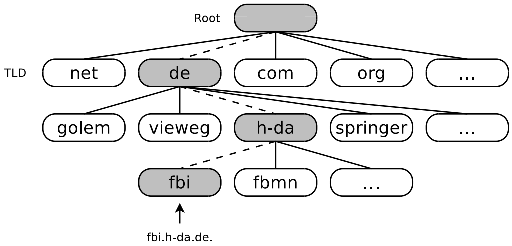
DNS replaced the local domain name tables in the config file `/etc/hosts`[1], which until then had been used for managing the domain names/IP addresses mappings. This file can still be used to override results retrieved by the DNS.

[1] On Windows systems: `%WINDIR%\system32\drivers\etc\hosts`

Domain Namespace (1/2)

- The domain namespace consists of a tree of domain names
  - Leaves and nodes are called labels
  - Each subtree is a domain
- A complete domain name consists of the concatenation of all labels of a path
- For the labels of the nodes right below the root ($\rightarrow$ Top Level Domains (TLDs)) only alphanumeric characters and hyphen (-) are allowed
  - The length of a label must be at least 1 and can be up to 63 characters
  - Labels must not start or end with a hyphen
  - It must not be all numeric
  - Each label ends with a period
- Domain names end with a period
  - The period is usually omitted, but from a formal perspective, a complete domain name – Fully Qualified Domain-Name (FQDN) ends with a period
- Examples for a complete domain name are `www.riot-os.org.` and `teaching.dahahm.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-Nameserver

http://www.root-servers.org (January 2022)

- The 13 root name 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
- Root servers do not consist of a single, but multiple physical servers, which are connected to a logical server
  - These hosts are located at different locations around the world and can be reached via anycast using the same IP address

Name	IPv4 address	IPv6 address	Location	Sites	Operator
A	198.41.0.4	2001:503:ba3e::2:30	distributed (Anycast)	16	Verisign, Inc.
B	199.9.14.201	2001:500:200::b	distributed (Anycast)	6	Information Sciences Institute
C	192.33.4.12	2001:500:2::c	distributed (Anycast)	12	Cogent Communications
D	199.7.91.13	2001:500:2d::d	distributed (Anycast)	168	University of Maryland
E	192.203.230.10	2001:500:a8::e	distributed (Anycast)	254	NASA Ames Research Center
F	192.5.5.241	2001:500:2f::f	distributed (Anycast)	289	Internet Systems Consortium (I
G	192.112.36.4	2001:500:12::d0d	distributed (Anycast)	6	Defense Information Systems A
H	198.97.190.53	2001:500:1::53	distributed (Anycast)	8	U.S. Army Research Lab
I	192.36.148.17	2001:7fe::53	distributed (Anycast)	68	Netnod
J	192.58.128.30	2001:503:c27::2:30	distributed (Anycast)	118	Verisign, Inc.
K	193.0.14.129	2001:7fd::1	distributed (Anycast)	79	RIPE NCC
L	199.7.83.42	2001:500:9f::42	distributed (Anycast)	196	ICANN
M	202.12.27.33	2001:dc3::35	distributed (Anycast)	7	WIDE Project

Structure of the DNS Database and the Resource Records

- The zone files contain lists of resource records (RR)
- Every RR is a name/value binding
- Every RR consists of 5 elements
  `<Name, Value, Type, Class, TTL>`
- Each name server may cache these entries in accordance to their TTL
- The table contains some types of RRs

Type	**Description**
`NS`	Specifies the name server which is responsible for the zone
`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 (*canonical*) name for a specific host
`MX`	Assigns the responsible mail server to a name.[2]
`PTR`	Provides the domain name associated with an IP address (for *DNS reverse lookups*).

[2] All other services use `CNAME`, `A` and `AAAA` resource records for the name resolution.

Example of a Domain Name Resolution (1/5)

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

```
dig +trace +additional -t A www.frankfurt-university.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

- For resolving the IP, 4 name servers have to be consulted one by one

The output of `dig` on the following slides contains several DNSSEC Resource Records (RR). DNSSEC provides authenticity and integrity of DNS data

- `RRSIG` = Signature Resource Record = Digital signature of a DNS Resource Record Set
- `NSEC3` = Hashed next secure entry within the zone (*chain-of-trust*)
- `DS` = Delegation Signer = Used to concatenate DNSSEC-signed zones. This way, several DNS zones are combined into a chain-of-trust and can be validated with a single public key

Example of a Domain Name Resolution (2/5)

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

; <<>> DiG 9.16.23 <<>> +trace +additional -t A www.frankfurt-university.de.
;; global options: +cmd
.      499597   IN   NS   a.root-servers.net.
.      499597   IN   NS   c.root-servers.net.
.      499597   IN   NS   j.root-servers.net.
.      499597   IN   NS   g.root-servers.net.
.      499597   IN   NS   b.root-servers.net.
.      499597   IN   NS   f.root-servers.net.
.      499597   IN   NS   m.root-servers.net.
.      499597   IN   NS   k.root-servers.net.
.      499597   IN   NS   i.root-servers.net.
.      499597   IN   NS   h.root-servers.net.
.      499597   IN   NS   l.root-servers.net.
.      499597   IN   NS   d.root-servers.net.
.      499597   IN   NS   e.root-servers.net.
.      503019   IN   RRSIG NS 8 0 518400 20220202050000 20220120...
...
;; Received 1125 bytes from 10.2.0.1#53(10.2.0.1) in 3 ms
```

- The final line contains the IP address 10.2.0.1 of the name server of the requesting host
  - This name server knows the IP addresses of the root name servers
  - IP addresses of 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.
de.      86400 IN  DS  26755 8 2 F341357809A5954311CCB82ADE114C6C...
de.      86400 IN  RRSIG DS 8 1 86400 20220202050000 2022012004...
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
s.de.net.   172800  IN  AAAA  2003:8:14::53
z.nic.de.   172800  IN  AAAA  2a02:568:fe02::de
;; Received 761 bytes from 198.97.190.53#53(h.root-servers.net) in 123 ms
```

- From the 13 root name servers, `h.root-servers.net` was randomly chosen, to send it the request for `www.frankfurt-university.de.`
- The reply contains 6 name servers responsible for the zone `.de.` to choose from

Example of a Domain Name Resolution (4/5)

```
frankfurt-university.de. 86400  IN  NS  deneb.dfn.de.
frankfurt-university.de. 86400  IN  NS  medusa.fh-frankfurt.de.
tjlb7qbojvmlf1s6gdriru7vsms1lg16.de. 7200 IN NSEC3 1 1 15 CA12B74ADB90591A TJLF... NS SOA
    RRSIG DNSKEY NSEC3PARAM
7blnr7smbefem25dg5q217hsnrlb5gg0.de. 7200 IN NSEC3 1 1 15 CA12B74ADB90591A 7BLP... A RRSIG
tjlb7qbojvmlf1s6gdriru7vsms1lg16.de. 7200 IN RRSIG NSEC3 8 2 7200 20220203123110 2022...
7blnr7smbefem25dg5q217hsnrlb5gg0.de. 7200 IN RRSIG NSEC3 8 2 7200 20220203123110 2022...
medusa.fh-frankfurt.de. 86400 IN  A 192.109.234.209
;; Received 629 bytes from 81.91.164.5#53(f.nic.de) in 13 ms
```

- From the 6 name servers in the reply, `f.nic.de` has been randomly chosen, to send it the request for `www.frankfurt-university.de.`
- The reply contains 2 name servers responsible for the zone `.frankfurt-university.` to choose from

Example of a Domain Name Resolution (5/5)

```
frankfurt-university.de. 86400  IN  NS  medusa.fh-frankfurt.de.
frankfurt-university.de. 86400  IN  NS  deneb.dfn.de.
;; Received 162 bytes from 192.76.176.9#53(deneb.dfn.de) in 16 ms
```

- From the 2 name servers in the reply, `deneb.dfn.de` has been randomly chosen, to send it the request for `www.frankfurt-university.de.`
- Result: The IP of `www.frankfurt-university.de.` is 192.109.234.218

The DNS protocol

- Queries may processed iteratively or recursively
- The maximum length of a DNS reply via UDP is 512 bytes

Agenda

- DNS
- Remote Shells
- HTTP
- E-Mail
- More Protocols

Telnet (Telecommunication Network)

- Protocol for remote access to a host in the network
- Specified in RFC 854
- Uses TCP via port 23
- Character-oriented → command line interface
- Software, which implements the protocol, is also simply called Telnet
- In the early versions the programs rsh and rlogin served a similar purpose
- Drawback: No encryption per default! → Replaced by SSH
- The Telnet client is often used as a network debugging tool

```
telnet> open localhost
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Debian GNU/Linux bookworm/sid
murdock login: user
Password:
```

Secure Shell (SSH)

- Provides an secure channel between two hosts over a potentially insecure network
- Specified in RFCs 4250, 4251, 4252, 4253, and 4254
- Uses TCP via port 22
- Originally developed by Tatu Ylönen at the Helsinki University of Technology in 1995
- Version 2 of the protocol has been released in 1996
- The most popular implementation for the server and client is OpenSSH[3]

Features

- Any TCP/IP connection can be tunneled over SSH (port forwarding)
- 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

[3] `https://openssh.com`

Agenda

Hypertext Transfer Protocol (HTTP)

- 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
- Specified in RFC 1945, 2068, 7540, and many more
- Uses TCP via port 80 or 443 (HTTPS → HTTP over a secure channel)
- Originally developed by Roy Fielding, Tim Berners-Lee, and others at CERN from 1989 onwards
- Currently in version 2 (HTTP/2) since 2015
- The proposed successor HTTP/3 is based on QUIC

World Wide Web

- Together with the concepts of URL[4] and HTML[5] it is the basis of the World Wide Web (WWW)
- Original main purpose: Loading web pages from webserver in a browser
- HTTP needs a reliable transport protocol → TCP
- Each HTTP message consists of:
  - *HTTP header*: Includes among others information about the encoding, desired language, browser, and content type
  - *Body*: Contains the payload, e.g., the HTML source code of a web page
- Today many application work on top of HTTP, e.g., using web sockets

[4]URL = Uniform Resource Locator

[5]HyperText Markup Language

HTTP Methods

- The HTTP protocol provides several requests messages

Request	**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.

HTTP Responses

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

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 $\Longrightarrow$ error was caused by server

Common HTTP Status Codes

Source: http.cat, Author: Tomomi Imura

- 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 Requests

- If an URL is accessed via HTTP (e.g., `http://example.teaching.dahahm.de/index.html`, the request for the resource `/index.html` is transmitted to the computer with hostname `example.teaching.dahahm.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 /index.html HTTP/1.1
Host: example.teaching.dahahm.de
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:96.0) Gecko/20100101 Firefox/96.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q
    =0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
...
```

Virtual Hosts (vhosts)

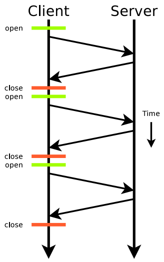
One server handles typically more than one domain, i.e., the same web server application may deliver multiple web pages at the same IP address for different domain names.

HTTP Response

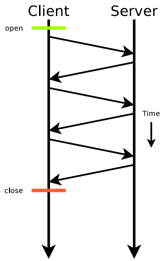
- The HTTP response of the web server consists of a message header and the message body with the actual message
  - In this case, the message body contains the content of the requested file `index.html`

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Fri, 28 Jan 2022 18:05:47 GMT
Content-Type: text/html
Content-Length: 274
Last-Modified: Fri, 28 Jan 2022 17:55:45 GMT
Connection: keep-alive
ETag: "61f42e21-112"
Accept-Ranges: bytes
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Example Page for teaching computer networks</title>
  </head>
  <body>
    <p>Happy networking!</p>
  </body>
</html>
```

HTTP Protocol Versions (HTTP/1.0 and HTTP/1.1)



- HTTP/1.0 (RFC 1945): Prior to any request, a new TCP connection is established and closed by default by the server after the transmission of the reply



- HTTP/1.1 (RFC 2616): By default, no connection termination is done
  - So the connection can be reused for multiple requests
  - Interrupted transmissions can be resumed with `HTTP/1.1`

HTTP Protocol Versions (HTTP/2)

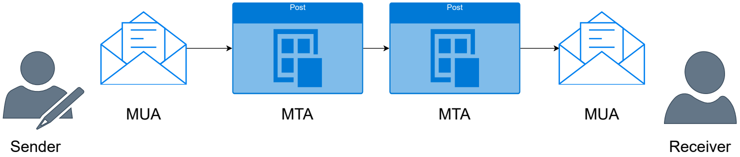
- HTTP/2 (RFC 7540): Changes from a text-based protocol to a binary one
  - Accelerates the data transfer by compressing the header with the *HPACK* algorithm (RFC 7541)
  - Enables the aggregation (*Multiplex*) of requests and a server can send (*Server Push*) data automatically, which it expects the browser to request immediately
    - Examples of such data are CSS files (Cascading Style Sheets), which specify the layout of web pages, or script files
  - Currently used by approx. 45 % of all web servers
- HTTP/3: Is not yet an RFC
  - Based on QUIC
  - Currently used by approx. 20 % of all web servers, but not yet supported by all browsers

Agenda

- DNS
- Remote Shells
- HTTP
- E-Mail
- More Protocols

Email – Architecture and Services

- Originally specified in RFCs 871 and 872 in 1982
- Required components:
  - Mail User Agent (MUA) → *mail client*
  - Message Transfer Agent (MTA) → *mail server*
- An email is composed of an envelope, header, and the body



Encoding

The body may contain ASCII encoded plain text or text in different encodings and other content following the MIME (Multipurpose Internet Mail Extensions) specification. It is considered good practise to send text in ASCII only.

Essential Protocols

- Simple Mail Transfer Protocol (SMTP)
  - Allows for the exchange (i.e., sending) of mails and is used for the communication between MTAs
  - The most recent specification in RFC 5321
  - Uses TCP (default port: 25)
- Post Office Protocol (POP)
  - Can be used to retrieve (download) the emails for a user from the server
  - Uses TCP (default port: 110)
- Internet Message Access Protocol (IMAP)
  - Can also be used to retrieve the emails for a user from the server, but typically leaves a copy at the server
  - Uses TCP (default port: 143)

Spamming, Phishing, Spoofing

Many issues with email arose over time. . .

- Spam: Sending a bulk load of unsolicited mails
- Phishing: Trick the receiver into revealing sensitive information or pay money
- Spoofing: Faking the identity of the sender

Source: https://artandlogic.com

Additional Security Protocols and Formats

- Simple Authentication and Security Layer (SASL) is a security framework for authentication of users that can be used in combination with SMTP
- DomainKey Identified Mail (DKIM), Sender Policy Framework (SPF), and Domain-based Message Authentication, Reporting and Conformance (DMARC) are authentication methods to check the validity of a MTA to prevent spam and phishing emails
- Secure/Multipurpose Internet Mail Extensions (S/MIME) and Pretty Good Privacy (PGP) are standards for encryption and signing of emails

SMTP Commands and Replies

- SMTP is a plain text protocol, important commands are:

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 alive (avoids timeouts)
`QUIT`	Log out from the SMTP server

- A SMTP server replies to a command with a three digit reply code and an optional text

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

- Be careful when operating a SMTP server – there are many tripwires

MTA Software

Popular SMTP servers are among others Postfix, qmail, Exim, IBM Lotus Domino, or MS Exchange. The first important implementation was Sendmail.

Sending Emails via SMTP

```
$ nc sea-02.cit.frankfurt-university.de 25
220 sea-02.cit.frankfurt-university.de Fra-Uas Mail System
HELO applecore
250 sea-02.cit.frankfurt-university.de
MAIL FROM: <oliver.hahm@riot-os.org>
250 2.1.0 Ok
RCPT TO: <oliver.hahm@fb2.fra-uas.de>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
From: <oliver.hahm@riot-os.org>
To: <oliver.hahm@fb2.fra-uas.de>
Subject: Testmail
Date: Fri, 28 Jan 2022 16:02:05 +0100

Hello!

And goodbye.
.
250 2.0.0 Ok: queued as 02496DF41D_1F54EBDF
QUIT
221 2.0.0 Bye
```

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

Email Header

```
Return-path: <oliver.hahm@riot-os.org>
Envelope-to: oliver.hahm@fb2.fra-uas.de
Delivery-date: Mon, 31 Jan 2022 13:17:36 +0100
Received: from smart-mail02.cit.frankfurt-university.de ([194.95.81.233])
        by klopfer.dv.fh-frankfurt.de with esmtps
        (envelope-from <oliver.hahm@riot-os.org>)
        for oliver.hahm@fb2.fra-uas.de; Mon, 31 Jan 2022 13:17:36 +0100
Received: from sea-02.cit.frankfurt-university.de ([194.95.81.231])
        by smart-mail02.cit.frankfurt-university.de with esmtps  (TLS1.2) tls...
Received: from mail.stillroot.org (mail.stillroot.org [176.9.132.253]) ...
        for <oliver.hahm@fb2.fra-uas.de>; Mon, 31 Jan 2022 12:17:34 +0000
        (GMT) ...
X-Virus-Scanned: Debian amavisd-new at ba.stillroot.org ...
Received: from applecore.local.domain (unknown [194.95.83.45])
        by mail.stillroot.org (Postfix) with ESMTPSA id 75FEB40363
        for <oliver.hahm@fb2.fra-uas.de>; Mon, 31 Jan 2022 13:17:28 +0100
        (CET)
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed; d=riot-os.org; ...
Date: Mon, 31 Jan 2022 13:07:12 +0100
From: Oliver Hahm <oliver.hahm@riot-os.org>
To: oliver.hahm@fb2.fra-uas.de
Subject: Testmail
Message-ID: <YffQ8JklzFLaCJN6@applecore.local.domain>
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Disposition: inline
User-Agent: Mutt/2.1.5 (31b18ae9) (2021-12-30)
```

Agenda

- DNS
- Remote Shells
- HTTP
- E-Mail
- More Protocols

Message Queuing Telemetry Transport (MQTT)

- Specified by the Organization for the Advancement of Structured Information Standards (OASIS) since 1999
- Requires from the layer below that message are transmitted …
  - in correct order
  - without loss
  - bi-directionally

⇒ TCP is typically chosen as transport layer in TCP/IP networks

- MQTT-SN[6] is variant resource constrained networks with less requirements to the lower layer → can run over UDP
- Follows a publish-subscribe paradigm
  - Clients can publish messages for a certain topic to a broker
  - Clients can subscribe for a certain topic at the broker
  - Whenever a new message is published at the broker, it informs the subscribed clients

[6] SN = Sensor Networks

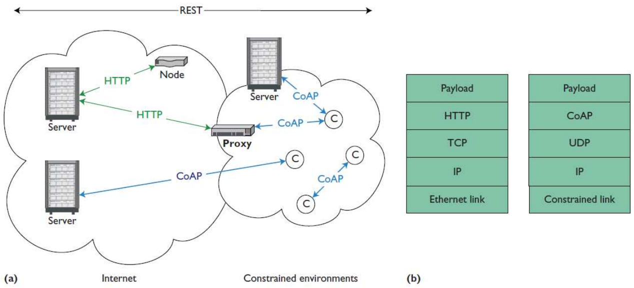
Signal Protocol

- Specified by the Signal Technology Foundation
- Originally developed as TextSecure Protocol by Trevor Perrin and Moxie Marlinspike in 2013
- Provides encrypted end-to-end communications
- Used by WhatsApp, Facebook Messenger, or Signal
- Uses phone numbers as identities

Constrained Application Protocol (CoAP)

- Protocol for constrained RESTFUL environments
- Specified in RFC 7252
- Uses UDP via port 5683
- Binary format

- Additional features allow for blockwise transfer, observe, or different transports
- Easy translation between CoAP and HTTP



Source: Bormann et al. 2012.

Message Types

- Requests
  - Non-confirmable
  - Confirmable
- Responses
  - Acknowledgement
  - Reset

You should now be able to answer the following questions:

- Do we need (standardized) application layer protocols for networking applications?
- How do you decide whether to use TCP or UDP on the transport layer for an application layer protocol?
- What are important application layer protocols in the Internet?
- What is a full-qualified domain name?
- What happens when you access a web page?
- How are emails delivered from the sender to the receiver?