# Willkommen bei Kommunikations- und Netztechnik!

– *Von Kupferkabel, Glasfaser und Mikrowelle*
*über Telefon, Ethernet und TCP*
*zu E-Mail, Webserver und REST.*

–

–

Heute: **E-Mail, Streaming, Datenverteilung**.

# Zusammenfassung Transportschicht I

- Transportschicht macht unzuverlässige Netzwerke zuverlässig
- TCP: Verbingung, UDP: unzuverlässiges Paket
- Verbindungsabbau: symmetrisch vs. asymmetrisch
- Adresse + Port definieren Endpunkt
- 2 Endpunkte definieren eine Verbindung
- Sequenznummern dürfen innerhalb der maximalen Paketlebenszeit nicht wiederholt werden
- 3 Way Handshake beim Verbindungsaufbau
- 4 Way Verbindungsabbau (auch 3 Way, da FIN+ACK in einem Segment gesendet werden dürfen)

# Zusammenfassung Transportschicht II

- Crash Recovery mit unterschiedlichen Client- und Serverstrategien

# Zusammenfassung Transportschicht III

- Kriterien von Congestion Control Algos:
  - Fairness
  - Effizienz
  - Konvergenz
- Definition Max Min Fairness: Bandbreite eines Flows kann nicht erhöht werden ohne Bandbreite eines anderen Flows zu senken, dessen Bandbreite nicht größer ist
- AIMD für Überlastkontrolle
- Slow Start und Fast Retransmission
- Sliding Window

# Ablauf heute

- Übungsaufgaben
- E-Mail
- DNS (Vorlesung 7)
- HTTP 2
- Rückblick

--- PAUSE ---

- Streaming
- Datenverteilung

# Ziele heute

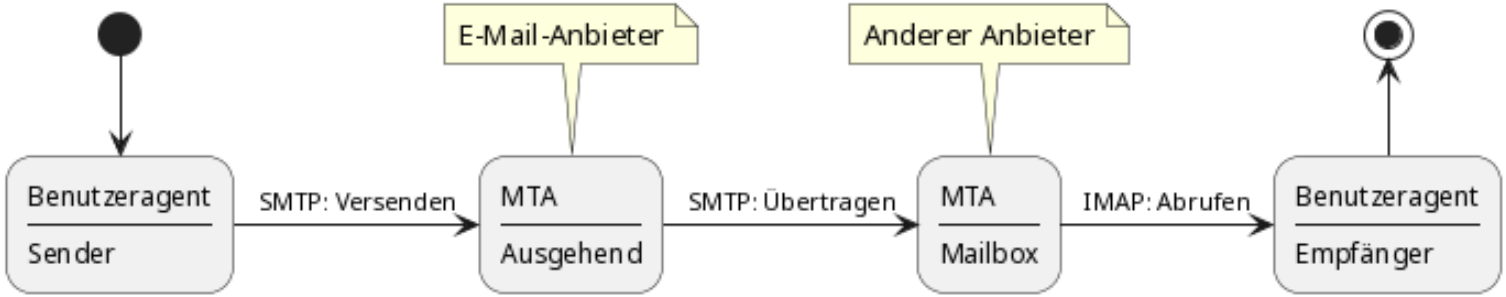
- Sie können die Struktur des E-Mail-Transports erklären
- Sie können SMTP, IMAP und MIME erkennen
- Sie können SMTP Umschlag (envelope) und Nachricht (message) unterscheiden
- Sie können die Kompressionsraten von Medien-Codecs einordnen
- Sie kennen übliche Optimierungen für Video-Codecs
- Sie können Unterschiede in den Anforderungen bei Video-on-Demand und Interaktiven Konferenzen erklären
- Sie verstehen den Nutzen von CDNs und Proxies
- Sie wissen, wodurch Peer-to-Peer-Netzwerke selbstskalierend sind

# E-Mail

- Verteiltes Protokoll
- Anbieterunabhängig
- Kostenlos verfügbar
- Notfallplan für den Zugang zu fast allen Diensten[1]

[1] Fast alle: Auf Handies wird E-Mail teils durch die (kostenpflichtige) Handynummer ersetzt.

# Architektur

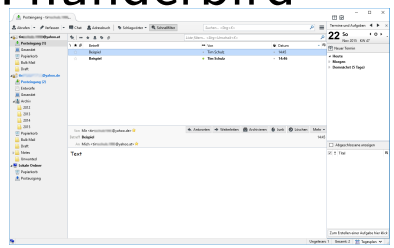



MUA Mail User Agent

MTA Mail Transfer Agent

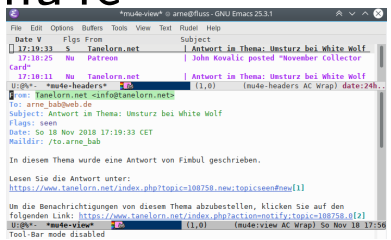
# E-Mail-Programme: Mail User Agent (MUA)

## Thunderbird

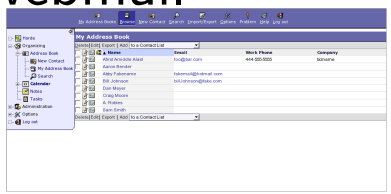


Tim Schulz
commons.wikimedia.org/wiki/File:Mozilla-Thunderbird-Windows.png

## mu4e

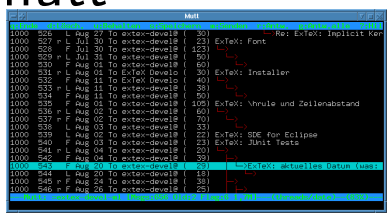


## webmail



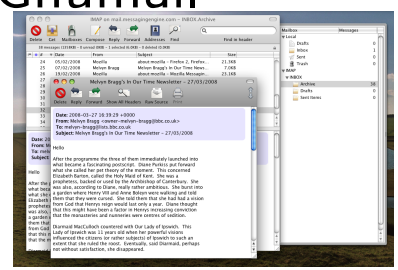
The Horde Project
commons.wikimedia.org/wiki/File:Horde-turba.png

## mutt



plaicy
commons.wikimedia.org/wiki/File:Mutt-Threaded-Modus.png

## Gnumail



Russ h commons.wikimedia.org/wiki/File:Gnumail1.2.0.png

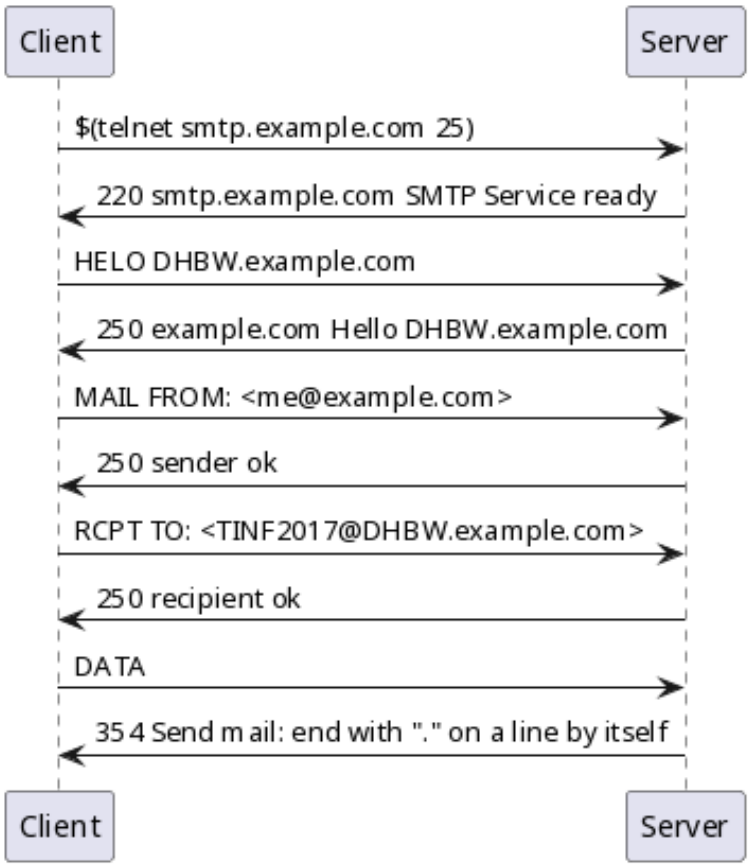
## hg email

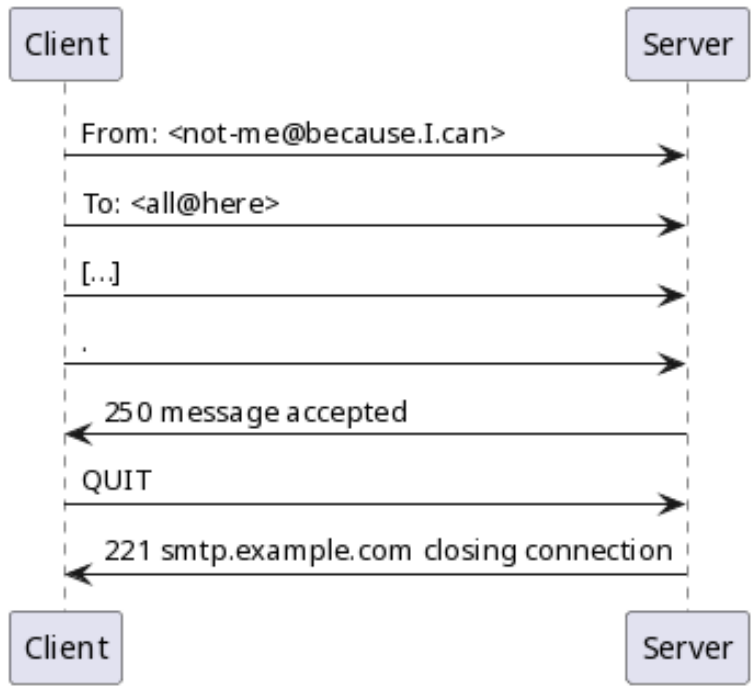
```
hg email --confirm --to USER@ANBIET
Diese Patch-Serie besteht aus 1 Pat

Cc:

Engültige Zusammenfassung:

From: USER@ANBIETER.TLD
To: USER@ANBIETER.TLD
Subject: [PATCH] Struktur der Vermi
 netzwerke/folien-6-netzanwendunger
 1 Dateien verändert, 107 Zeilen hi

Sicher, dass Sie jetzt senden möcht
```

# Mobile Clients?

- Kennen Sie gute? Frei? Offlinefähig?
- Hochleistung nötig!
  - 150k Nachrichten üblich
  - 15 GiB an Daten auch üblich
  - Interaktiver Zugriff nötig!
- Eudora immernoch unerreicht. computerhistory.org/blog/the-eudora-email-client-source-code/

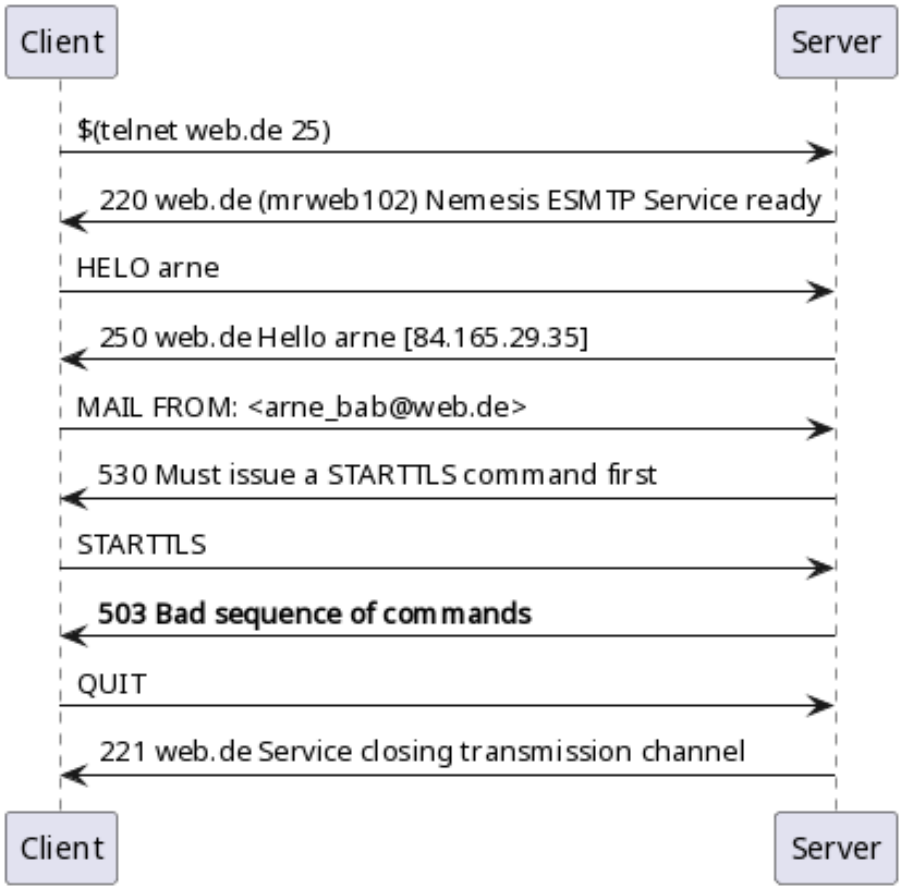
# SMTP




`example.com`: explizit verbotene Domain.

# SMTP mit Telnet in Wirklichkeit




*HELO erlaubt kein STARTTLS! → EHLO*

# SMTP, in Wirklichkeit

```
220 web.de (mrweb003) Nemesis ESMTP Service ready
250-web.de Hello fluss [84.165.29.35]
250-8BITMIME
250-AUTH LOGIN PLAIN
250-SIZE 141557760
250 STARTTLS
220 OK
250-web.de Hello fluss [84.165.29.35]
250-8BITMIME
250-AUTH LOGIN PLAIN
250 SIZE 141557760
AUTH PLAIN <omitted>
235 Authentication succeeded
MAIL FROM:<arne_bab@web.de> SIZE=3111
250 Requested mail action okay, completed
RCPT TO:<arne_bab@web.de>
250 OK
RCPT TO:<carlo.goetz@gmail.com>
250 OK
DATA
354 Start mail input; end with <CRLF>.<CRLF>
...
.
250 Requested mail action okay, completed: id=0McFQF-1
QUIT
221 web.de Service closing transmission channel

Process smtpmail connection broken by remote peer
```

*(aus dem Buffer **trace of SMTP session to smtp.web.de** aus meinem Emacs)*

# IMAP

- Internet Message Access Protocol (RFC 3501)
- Warum nicht POP3 (Post Office Protocol 3)?
  - Dimensionierung der Server: IMAP: gleichmäßige Last. POP3 alle Last beim Start des E-Mail-Programms.
  - IMAP kann E-Mails auf dem Server verwalten.
    - Durchsuchbare Ordner (mailboxes)
    - Mehrere Endgeräte
      IMAP war schon Cloud vor der Cloud (1992)

# IMAP Protokoll

```
2.      Protocol Overview

2.1.    Link Level

The IMAP4rev1 protocol assumes a reliable data stream such as
that provided by TCP.  When TCP is used, an IMAP4rev1 server
listens on port 143.
```

tools.ietf.org/html/rfc3501#section-2

# IMAP praktisch

*Ausgaben von Telnet mit .., Antworten mit Präfix –*

```
$ telnet imap.web.de 143
.. Trying 212.227.17.162...
.. Connected to imap.web.de.
.. Escape character is '^]'.
-- * OK [CAPABILITY IMAP4rev1 CHILDREN ENABLE ID IDLE LIST-
A0001 NOOP
-- A0001 OK NOOP completed
A0002 LOGOUT
-- * BYE Server logging out
-- A0002 OK LOGOUT completed
.. Connection closed by foreign host.
```

# ONLINE-PAUSE

10 Minuten Pause.

# Format von E-Mails

- Envelope
- Header
- Leerzeile
- Daten
  - MIME

# Format: Umschlag vs. Nachricht I

*„Was ich mindestens erwarte, ist den Unterschied zwischen Envelope und Header zu verstehen.“* — Backend-Entwickler bei web.de/gmx

**2.3.1. Mail Objects** tools.ietf.org/html/rfc5321

SMTP transports a mail object. A mail object contains an envelope and content.

The SMTP envelope is sent as a series of SMTP protocol units (described in Section 3). **It consists of an originator address** (to which error reports should be directed), **one or more recipient addresses**, and optional **protocol extension** material. Historically, variations on the reverse-path (originator) address specification command (MAIL) could be used to specify alternate delivery modes,

# Format: Umschlag vs. Nachricht II

such as immediate display; those variations have now been deprecated (see Appendix F and Appendix F.6).

The SMTP content is sent in the SMTP DATA protocol unit and has two parts: the **header section** and the **body**. If the content conforms to other contemporary standards, the header section consists of a collection of **header fields**, each consisting of a **header name**, a **colon**, and **data**, structured as in the message format specification (RFC 5322); the body, if structured, is defined according to **MIME** (RFC 2045). The content is textual in nature, expressed using the US-ASCII repertoire. Although SMTP extensions (such as "8BITMIME", RFC 1652) may relax this restriction for the content body, the **content header fields are always encoded using the US-ASCII** repertoire. Two MIME extensions (RFC 2047 and RFC 2231) define an algorithm for

# Format: Umschlag vs. Nachricht III

representing header values outside the US-ASCII repertoire, while still encoding them using the US-ASCII repertoire.

Kurz: Das einzig Verlässliche ist:

- **Envelope To**.

Wäre das falsch, hätte die E-Mail Sie nicht erreicht.

Envelope From ist im Umschlag, aber nicht gesichert.

Alles, das keine explizite Kodierung gesetzt hat, ist 7 Bit US-ASCII.

# Format Beispiel

```
Return-Path: <USER@ANBIETER.TLD>
Received: from pop3.web.de [212.227.17.177]
    by localhost with POP3 (fetchmail-6.3.26)
    for <arne@localhost> (single-drop); Sun, 18 Nov 2018 19:00:12 +0100 (CET)
...
Received: from fluss.draketo.de ([84.165.29.35]) by smtp.web.de (mrweb003
 [213.165.67.108]) with ESMTPSA (Nemesis) id 0Lylol-1fLj8s39GB-0167Yi for
 <USER@ANBIETER.TLD>; Sun, 18 Nov 2018 18:45:57 +0100
MIME-Version: 1.0
Content-Type: text/plain; charset="utf-8"
Content-Transfer-Encoding: base64
Subject: [PATCH] Struktur der Vermittlungsschicht
Message-Id: <201d4a8e31a16734b10b.1542563152@fluss.Baez>
X-Mercurial-Node: 201d4a8e31a16734b10b91a9da5bd52ddda5a06b
...
User-Agent: Mercurial-patchbomb/4.7.1
Date: Sun, 18 Nov 2018 18:45:52 +0100
From: USER@ANBIETER.TLD
To: USER@ANBIETER.TLD
X-Spam-Flag: NO
Envelope-To: <USER@ANBIETER.TLD>
X-Spam-Flag: NO
X-UI-Filterresults: notjunk...

IyBIRyBjaGFuZ2VzZXQgcGF0Y2gKIyBVc2VyIEFybmUgQmFiZW5oYXVzZXJoZWlkZSA8YmFiQGRy
YWtldG8uZGU+CiMgRGF0ZSAxNTQyNTYyNjMzIC0zNjAwCiMgICAgICBTdW4gTm92IDE4IDE4OjM3
OjEzIDIwMTggKzAxMDAKIyBOb2RlIElEIDIwMWQ0YThlMzFhMTY3MzRiMTBiOTFhOWRhNWJkNTJk
...
```

# Format: MIME

- Multipurpose Internet Mail Extensions
- Wie verpacken Sie beliebige Inhalte in 7-Bit ASCII?
- äöüßéĉŝĝĥŭç

# Format: MIME Beispiel I

```
Message-ID: <87bm6mjkwy.fsf@web.de>
MIME-Version: 1.0
Content-Type: multipart/signed; boundary="=-=-=";
    micalg=pgp-sha256; protocol="application/pgp-signature"

--=-=-=
Content-Type: text/plain; charset=utf-8
Content-Transfer-Encoding: quoted-printable

Hi Carlo,
...

--=-=-=
Content-Type: application/pgp-signature; name="signature.asc"
```

# Format: MIME Beispiel II

```
-----BEGIN PGP SIGNATURE-----

iQIzBAEBCAAdFiEE801qEjXQSQPNItXAE++NRSQDw+sFAlvxsGEACgkQE++NRSQD
w+sqnxAAtpfgtJKJljrVf2WHIiDXB86TJJhkyQ7ufl7Ses3zRI1i28RCOOKfYVYe
6CTqOd9NaECn8S/rc2FetHcJnUpW3NwehPJWYyZtMnqFgFjI217acwFwC1T465XO
DW9xRXTPU3nJZSL2yk8OGvQGb6X+lgZn8wN7iuM+8HdOZF95fCA2iyi9/dkanS6U
Uvgi9W+qY+YeLjnsCq7qZcvIQeVizOWrVa4jgfPibwenVnOWx2WIeLGAGGhSUfc9
cTicwjYrZL5rHWT1DsR2NmjNP+fzuKprkZwwTJYhMVzc8z7ExN6nYAwzFWO2EFQa
qpECJEr/hxx8B0i8gRdmBli+AAYouUaLpzap1K2yzQWsfXXpsh0o5DGzdJ3ejv6z
Ts0+X4V2F+uiay9x2lP/agPXJK5IvyQ3lNAAPVY22eW/rToSOmltS/zf/uQxkQyw
SMVrkXMrCdUn6eQ++AMzvGFxliN6NEfJW3UviwDWE4U21C/PH2UVuf3xgCG5PNx+
TmPVn69UxsLKXdagqqWDUUAaBzyml3Dcilnw26tDo/3LAT6H8MWRhASPRpHS5SWV
axFMDRWOsFJsLLNrUC0BGGM6SnEECsZT/R72WHA1O1oIByMUTJaTESpl5nyTzo51
3of0/GifsOJtbZsu8IsZnHQmGVERB2Q381Gbkh8TIVNIqAZwZ/GIswQBAQgAHRYh
BN0ovebZh1yrzkqLHdzPDbMLwQVIBQJb8bBhAAoJENzPDbMLwQVI4DID/373p3W0
6YCAbdl1t0BMhZ03vRL7/fX43CW9oRYej3RJ10vNME2OEZmY+2G1DURDsA9mOXcL
39n1C6UCN/q/7VJXu7fSdqOYMlZFXfAjBJdWTZegJcF2AjS4gapgbkSnYjK6PYrS
dwKVjHoHdiASjPPdbmgL/HWpR8BahJiuR+mY
=ZvaC
-----END PGP SIGNATURE-----
--=-=-=--
.
```

# MIME: WTF?!?

In der vorletzten Folie war etwas schrecklich. Was war es?

# MIME-Typen

Kategorisierung von Inhalten. Beispiele:

| Typ | Endung | Bedeutung |
|---|---|---|
| application/octet-stream | - | Beliebige bytes |
| application/gzip | *.gz | GNU Zip |
| application/json | *.json | Javascript-Object-Notation |
| application/pdf | *.pdf | PDF-Dateien |
| audio/basic | *.au, *.snd | Sound-Dateien |
| audio/mpeg | *.mp3 | mp3-audio |
| image/png | *.png | PNG-Bilder |
| multipart/mixed | - | multipart ohne Bezug |
| text/plain | *.txt | Text |
| text/css | *.css | Style-Sheets |
| text/html | *.html | HTMI-Dateien (Webseiten) |

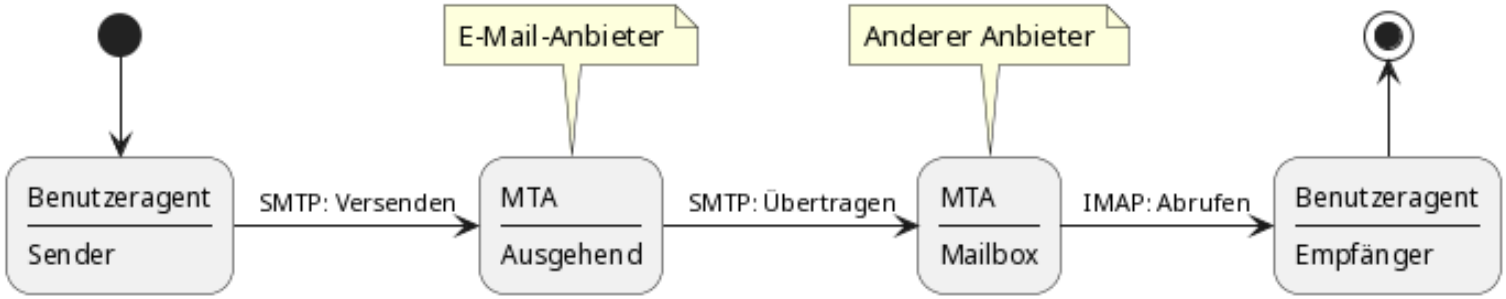
# Weitere E-Mail Dienste

- Mailinglisten: Ersetzen Header $\rightarrow$ möglich, da Header ungeprüft.
- Autoresponder + Urlaubsagent: Auf dem Server $\rightarrow$ Protokoll zwischen MUA und MTA.

# Weitere Themen

- Spam-Abwehr. Effektiv ungelöst: Serverside mit hohen Kosten. Spamassassin war mal gut. Spam-„Qualität“ wird immer besser. Kostenlose globale Sichtbarkeit ist schwierig.
  - WhatsApp entfernt das "kostenlos"durch den Zwang zur SIM-Karte.
  - Skalierende Lösungen durch Aufgabe globaler Erreichbarkeit möglich (eigene Arbeit).
    - www.draketo.de/english/freenet/deterministic-load-decentralized-spam-filter
  - Ende-zu-Ende Signaturen => mailvelope auf web.de/gmx
- Verschlüsselung: PGP → pEp (pretty easy privacy) + autocrypt: Automatische Verschlüsselung mit Sender-Identifikation und TOFU (trust on first use).
  - https://gnupg.org

# Zusammenfassung




- MUA: E-Mail-Client.
- MTA: Mail Transfer Agent
- SMTP: MUA → MTA oder MTA → MTA. text via telnet
  - **Umschlag**: MAIL FROM (ich), RCPT TO (Sie).
  - Einzig verlässlicher Wert: **Envelope To**.
  - EHLO: Extended HELO.
- IMAP: MTA → MUA, Textprotokoll, Port 143.
  - Serverseitige Operationen!

# Einwurf: Rückblick auf die Vorlesung

- 8 Gruppen: Fenster und Tür. Jede Reihe.
- Stichwörter sammeln und auf Zettel.
- Nach 5 Minuten:
  - Stichwörter von Tür zu Fenster und umgekehrt.
  - Wählen Sie eins per Zufall und besprechen Sie es.
- Nach 10 Minuten: 1 min Vorstellung pro Team

# PAUSE

# Streaming

Kopimismus

*Unsere Mission ist der ewige Kampf gegen die Entropie.*

Codecs

*. . . gegen **unsichtbare** Entropie. Die einzig gute Entropie ist die, deren Fehlen wir bemerken.*

# Themen

- Audio-Codecs
- Video-Codecs
- Bild-Codecs
- Stream via Download
- Interaktive Medien

# Audio-Codecs

- Rauschende 64kbit/s $\rightarrow$ kristallklare 16kbit/s
- Das Gehör verstehen
- Moderne Optimierungen
- Namen: MP3, AAC (MPEG4), Vorbis, Opus

# Bedeutung von Codecs

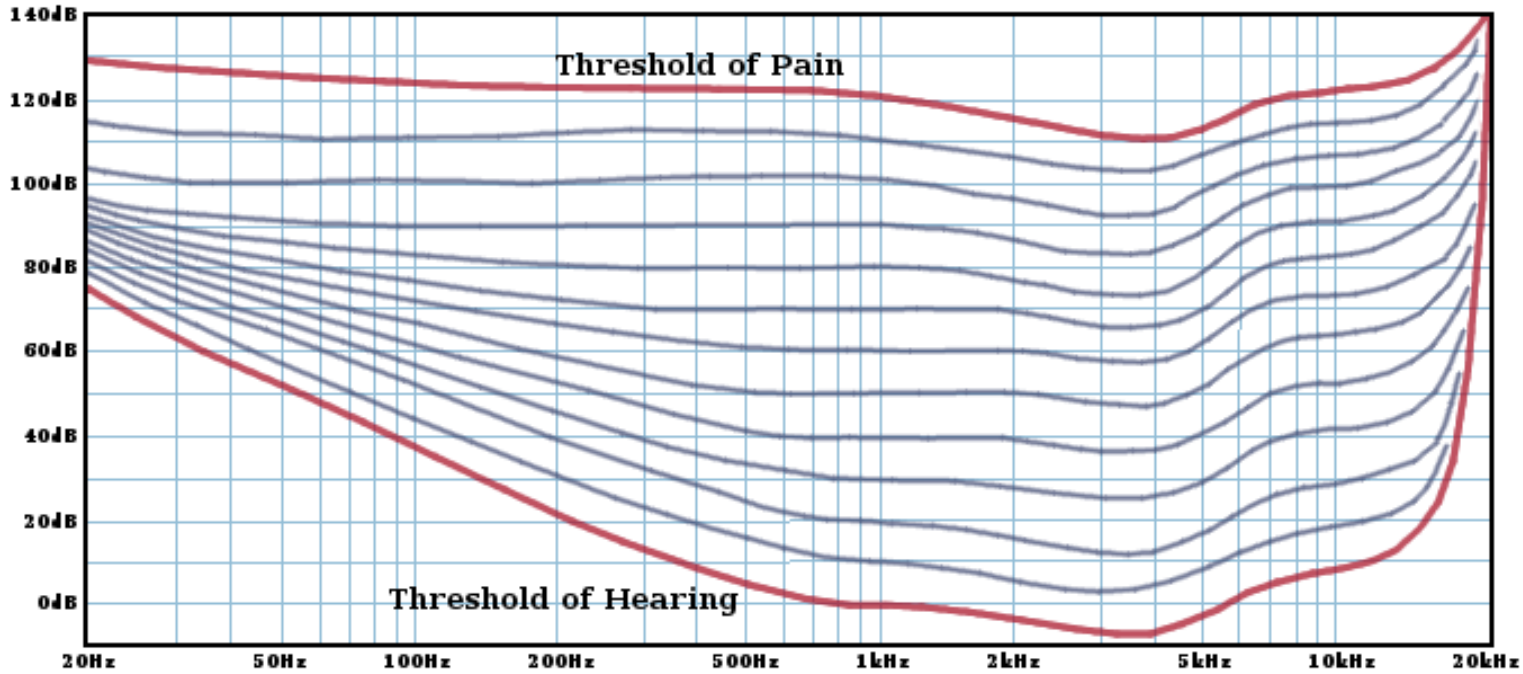
MP3 hat wirklich Musik revolutioniert:

- 128kBit/s ⇒ 600 MiB CD zu 60MiB.
- Große Festplatten 1998 hatten 47 GiB:[2]
  - 90 kopierte CDs
  - 900 als MP3 gerippt!
- Telefon: 64kBit/s → SILK/Opus 1.3: 9kBit/s![3]

*2019: LPCNet vocoder: 1.6kBit/s für Sprache!*[4]

[2] https://de.wikipedia.org/wiki/Festplattenlaufwerk
[3] How Opus came to be: https://jmvalin.dreamwidth.org/16616.html
[4] https://people.xiph.org/~jm/demo/lpcnet_codec/ — **öffnen**

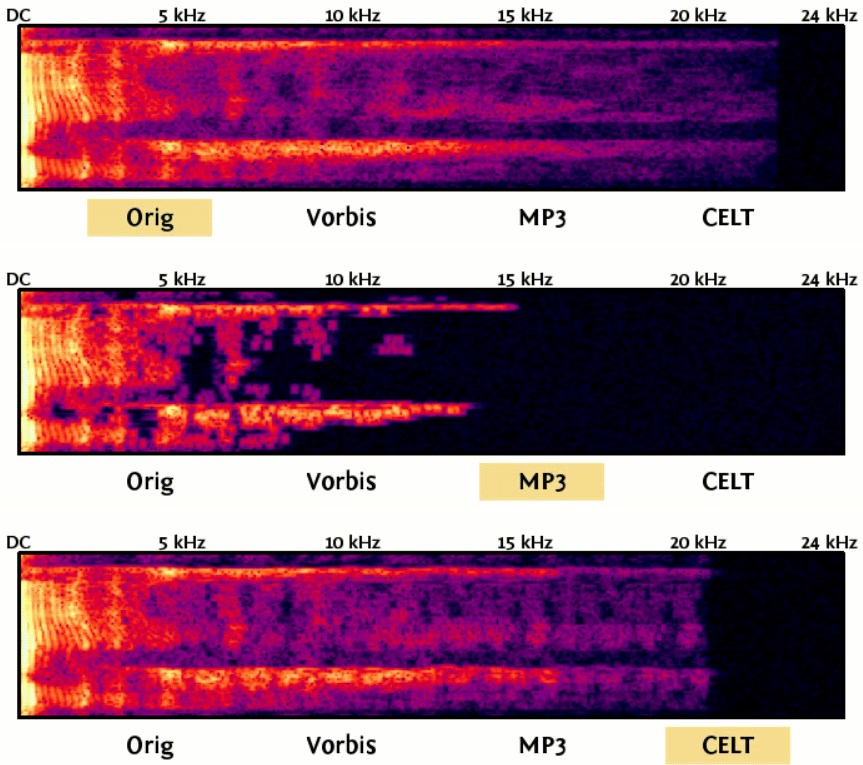
# Das Gehör verstehen




*„Neben“ lauten Tönen weniger!*

Dank Monty: people.xiph.org/~xiphmont/demo/neil-young.html

# Moderne Optimierungen




- orig: >1000 kbit/s
- mp3: 128 kbit/s
- CELT: 32 kbit/s
- SILK: 6-40 kbit/s
- LPCNet: 1,6 kbit/s (PDF)

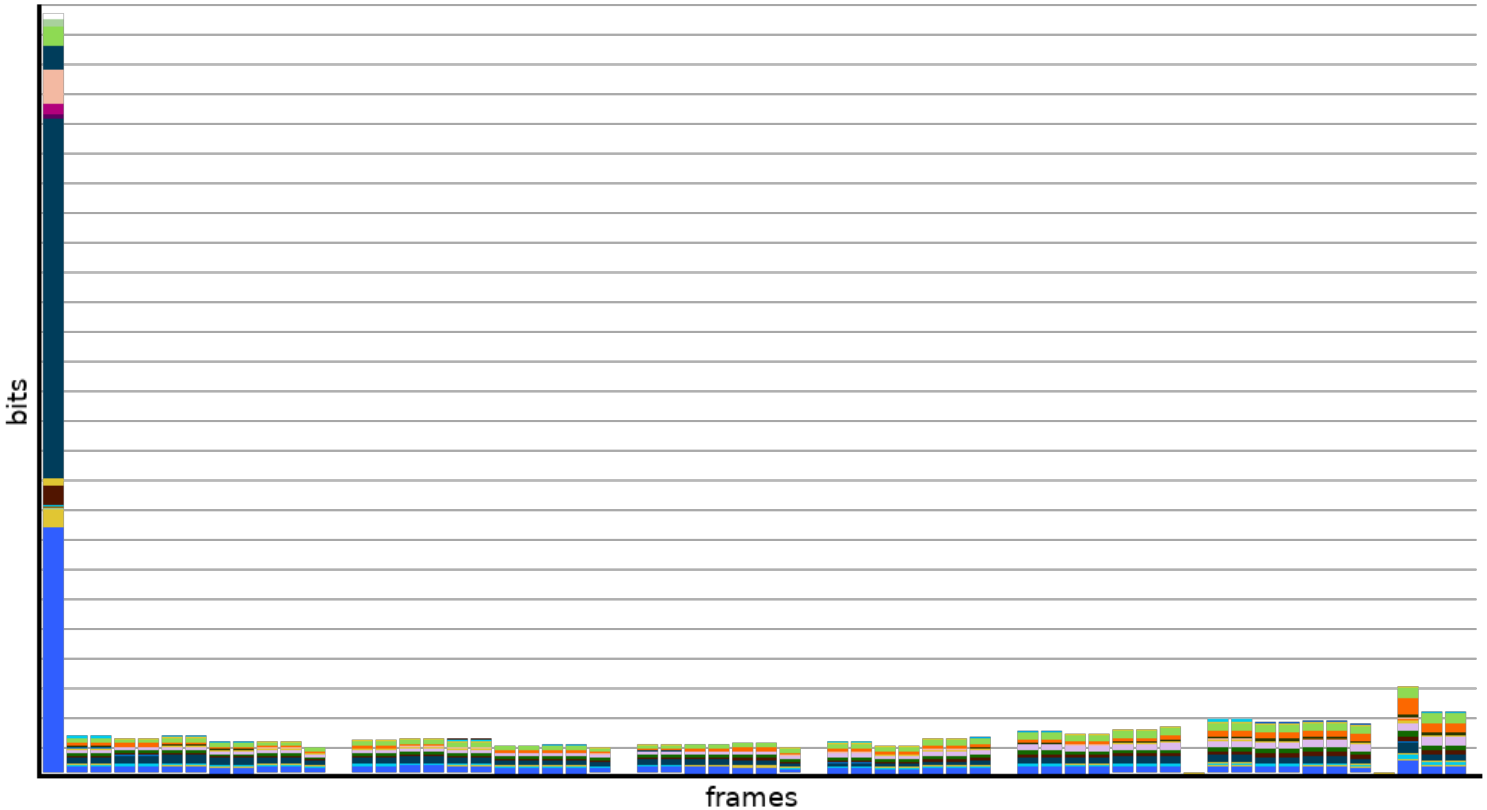
*SILK für Opus Sprache genutzt, Celt für Opus Musik:*
people.xiph.org/~xiphmont/demo/celt/demo.html
people.xiph.org/~jm/opus/opus-1.3/

# Video-Codecs

Video spart Faktor 100:

- MiniDV: 13GiB pro Stunde (29MBit/s)
  - MPEG1: 675MiB (1.5Mbit/s)
  - VP9: 90MiB (200kbit/s) — für Anime u.ä.
  - *AV1: 60MiB, wenig bewegt: 45MiB*

# Video-Codecs: Keyframes + Inkrementelle Änderungen




*Vergangenheit oder Vergangenheit + Zukunft*

https://people.xiph.org/~xiphmont/demo/av1/demo1.shtml

https://people.xiph.org/~xiphmont/demo/av1/demo2.shtml

# Vergleich: 132 MiB h264 vs. 7 MiB vp9 q62 (min: 64) I

h264, 132 MiB

# Vergleich: 132 MiB h264 vs. 7 MiB vp9 q62 (min: 64) II

vp9, 7 MiB

h264, 132 MiB

# Vergleich: 132 MiB h264 vs. 7 MiB vp9 q62 (min: 64) III

vp9, 7 MiB

# Vergleich: 132 MiB h264 vs. 7 MiB vp9 q62 (min: 64) IV

# Sichtbar: x264, vp9, av1

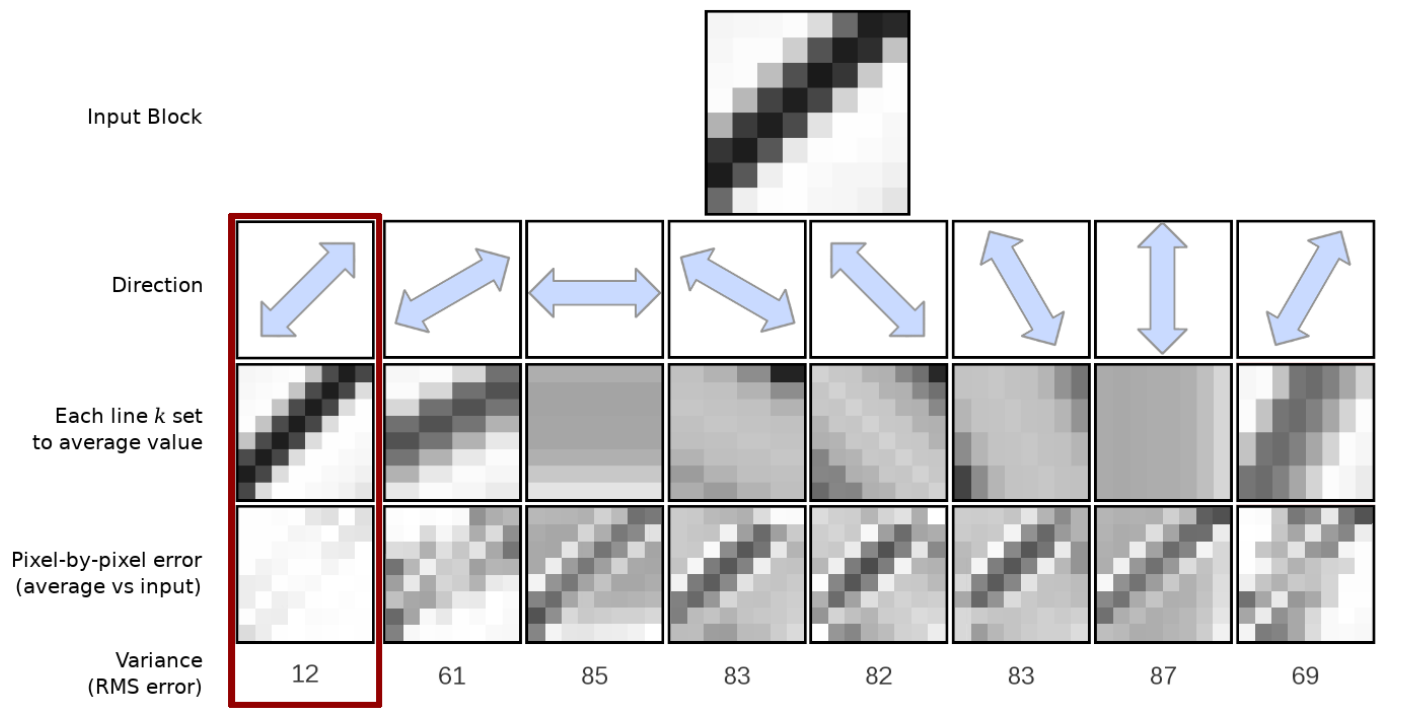
| 100kbit/s | 200kbit/s |
| --- | --- |
| x264_100k.mp4 | x264_200k.mp4 |
| vp9_100k.mp4 | vp9_200k.mp4 |
| av1_100k.mp4 | av1_200k.mp4 |

Video: Tears of Steel, (CC) Blender Foundation | mango.blender.org, https://creativecommons.org/licenses/by/3.0/

https://www.singhkays.com/blog/its-time-replace-gifs-with-av1-video/#encore

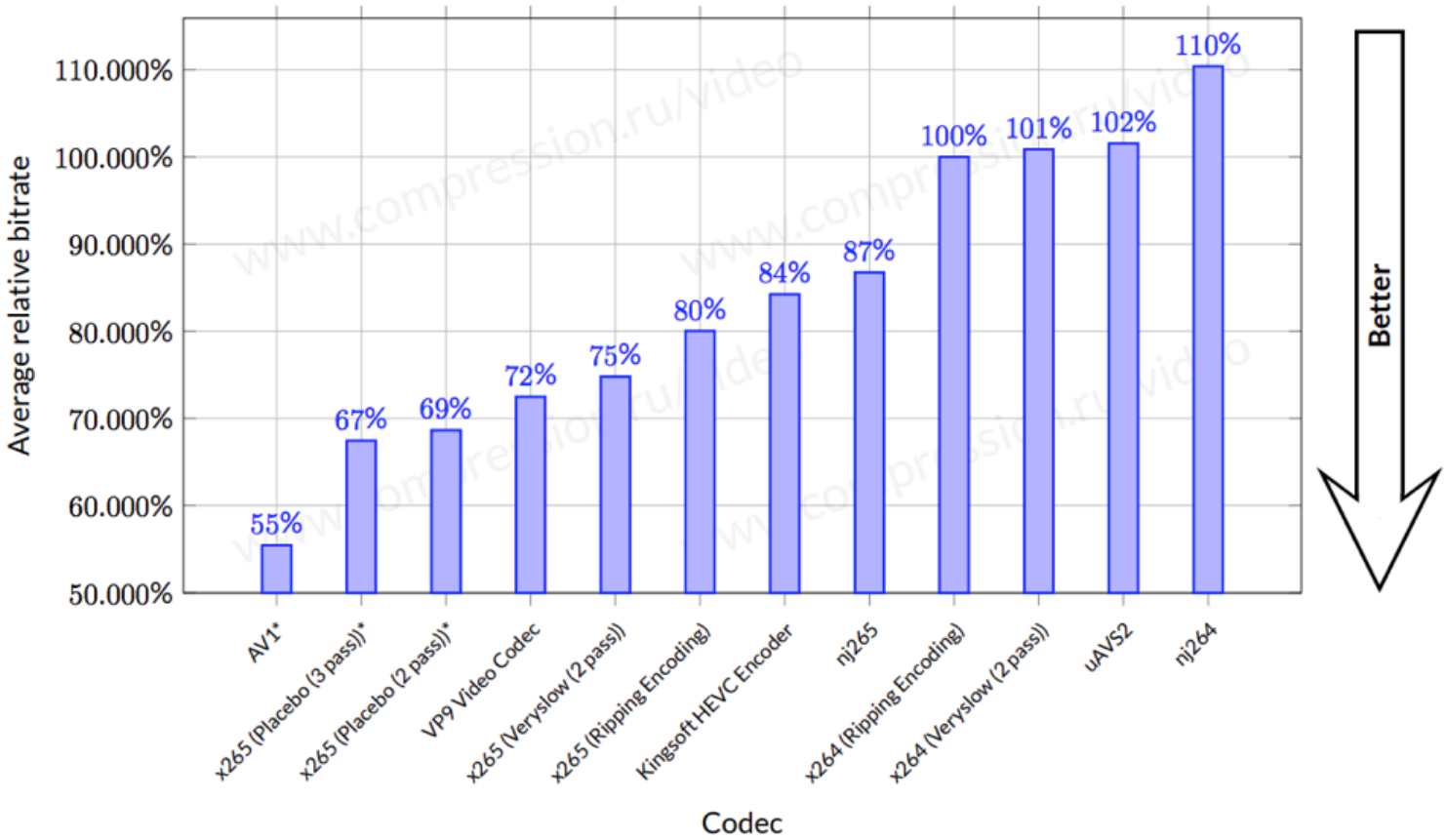
# Optimierungsbeispiel: Directional Paint, Konzept

# Optimierungsbeispiel: Directional Paint, Implementierung




CDEF: https://people.xiph.org/~xiphmont/demo/av1/demo2.shtml

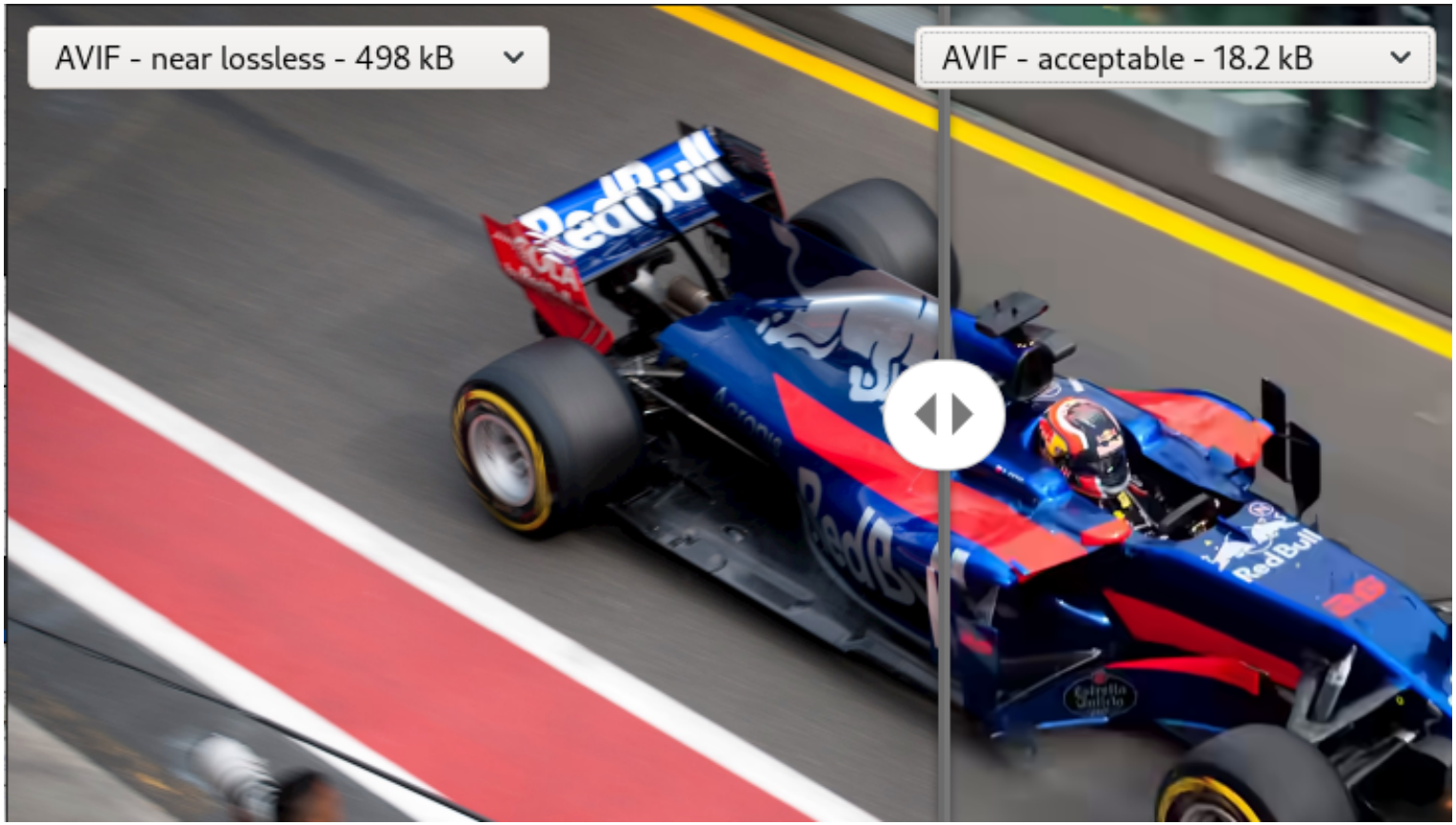
# Die letzten 15 Jahre: h264 zu AV1




https://blog.mozilla.org/blog/2018/07/11/royalty-free-web-video-codecs/ cc by-sa

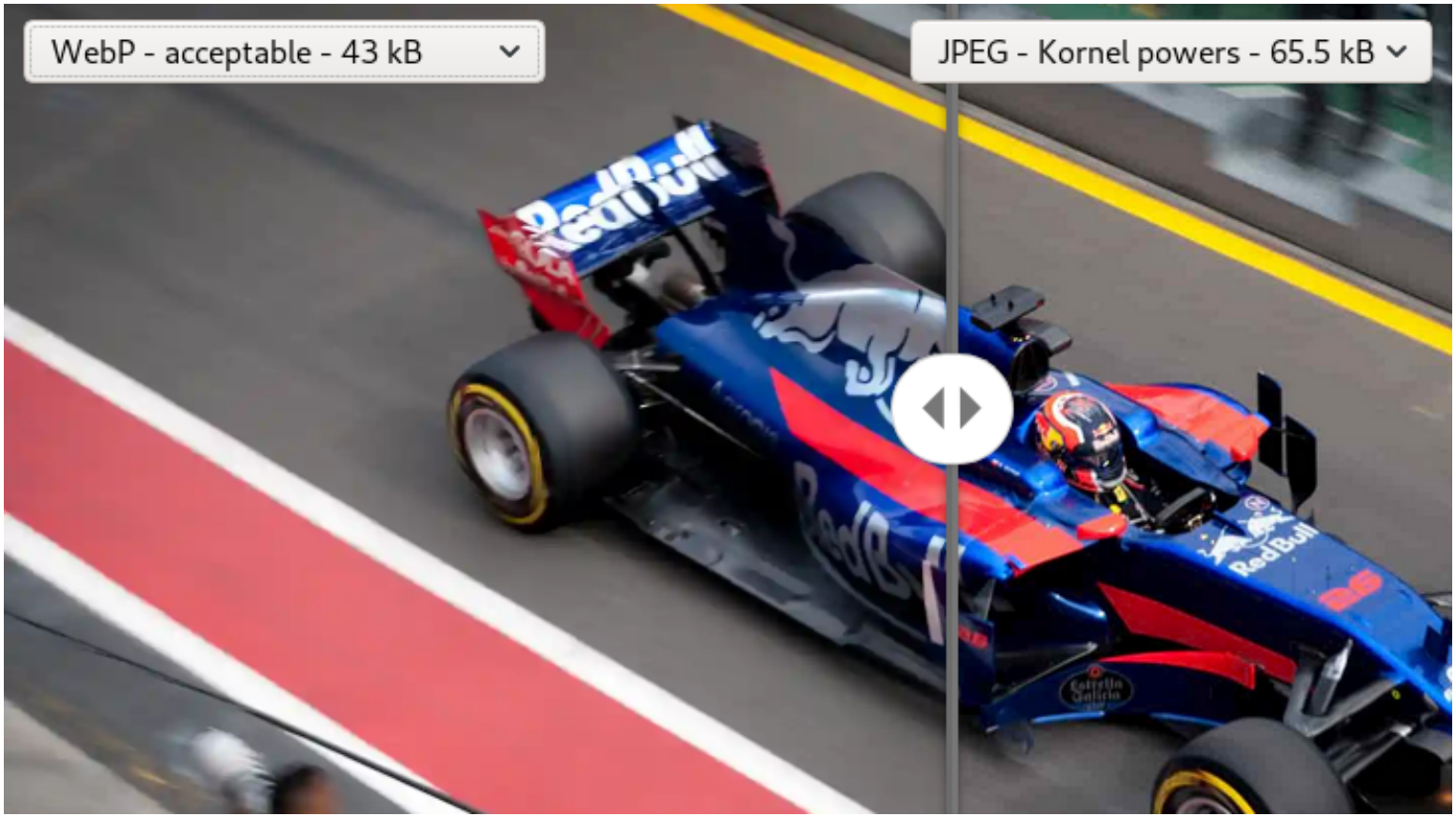
Tiefer einsteigen: http://aomanalyzer.org/

# Bild-Codecs: Beyond JPEG: lossless 498kB to AVIF 18.2kB

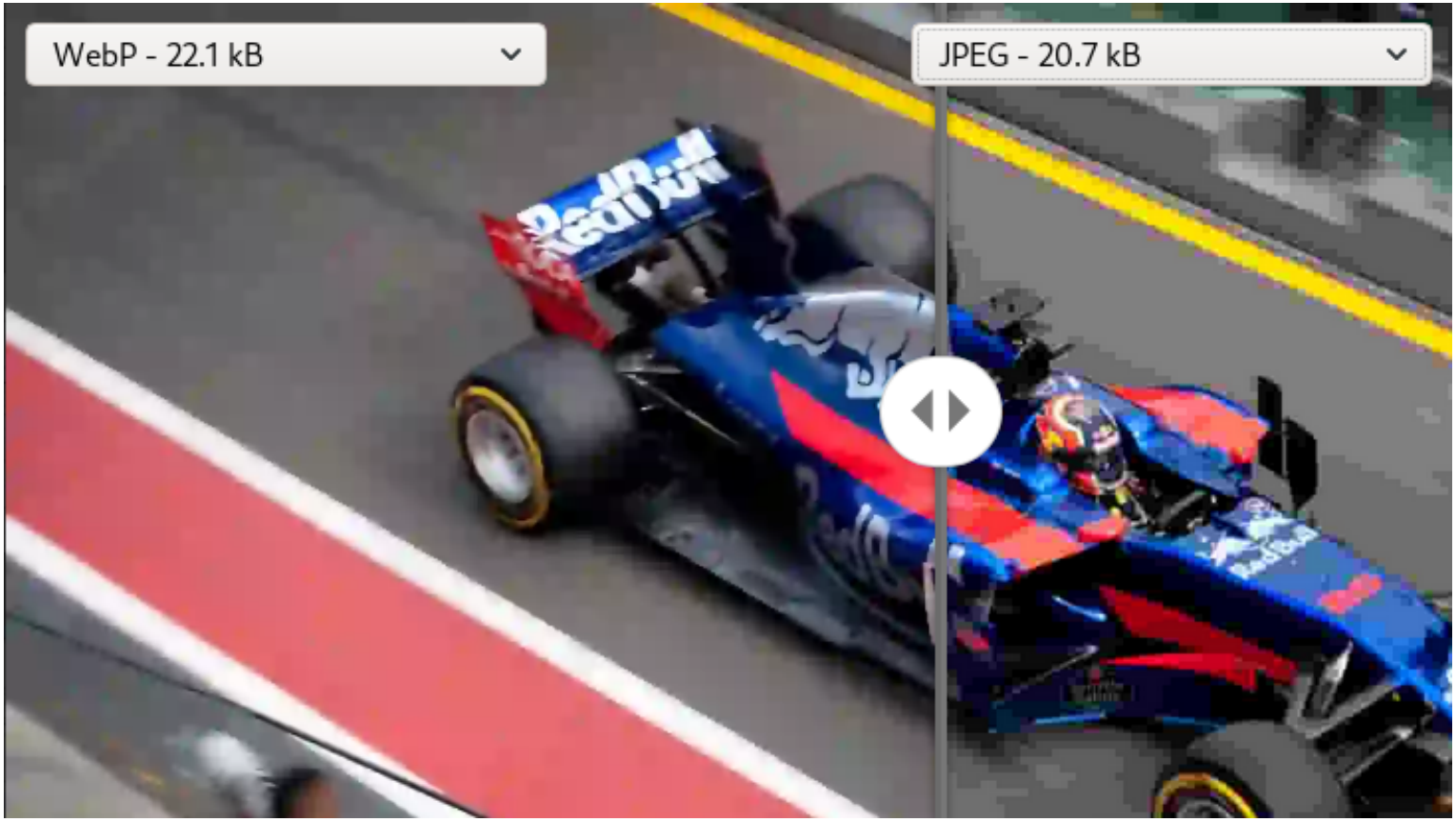



*Quelle (ich darf es als cc by-sa verwenden):*
*jakearchibald.com/2020/avif-has-landed*

# Bild-Codecs: Beyond JPEG: webp 43kB vs. JPEG 66kB



# Bild-Codecs: Beyond JPEG: webp 22kB vs. jpeg 21kB



# Zwischen-Zusammenfassung: Codecs

- Audio:
  - CD: 1333 kib/s
  - opus: 16 kib/s
  - LPCnet: 1,6 kib/s (Sprache)
  - **Faktor 85 – 850** → https://caniuse.com/opus
- Video:
  - dv uncompressed: ≈ 150.000 kib/s
  - dv compressed: ≈ 30.000 kib/s
  - av1: 100kib/s
  - **Faktor: 1700** → https://caniuse.com/av1
- Bild:
  - Original: 3500kB; 1920x1080 ⇒ 13.5bit/pixel

# ONLINE-PAUSE

10 Minuten Pause

# Unterschiedliche Anforderungen und Randbedingungen

- Video-On-Demand (VOD)
- Live-Streaming
- Interaktive Medien

→ *An der Tafel sammeln*

# Anwendung: Video on-demand

- Startverzögerung minimieren
- Zwischenspeicher minimieren
- Cache glättet Jitter (aber Youtube hängt -,-)
- Geringe Anforderungen: TCP reicht
- `<video>`-Tag macht das einfach, mit dem richtigen MIME-Typ
- Teil-Anfragen über Range-Header
- Einmal enkodiert, oft dekodiert.

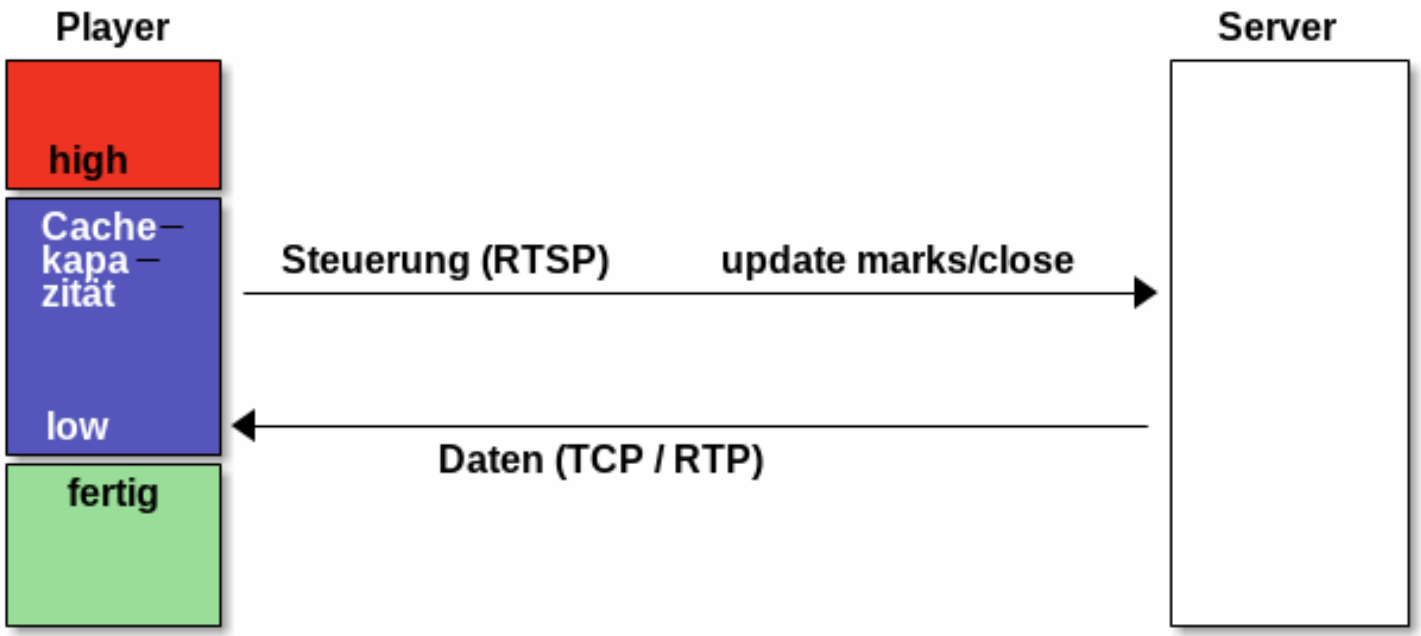
*Früher über Mediaplayer mit RTSP, heute macht der Browser alles.*

# Live-Streaming

- Keine Optimierung mit Daten aus der Zukunft
- Begrenzte Enkodierungs-Zeit
- Dafür: Multicast vom Provider
- Cache-Optimierung mit RTSP
- RTP: Real-Time Transport Protocol
  - UDP + Steuerpakete (RTCP: Real-Time Control Protocol)

# RTSP: Real-Time Streaming Protocol

- High-Water-Mark,
- Low-Water-Mark,
- Abbrechen
- ...



# Interaktive Medien; Herausforderung: Latenz minimieren

Round-Trip (2x Strecke), 2/3tel Lichtgeschw. (200 000 km/s)

- Karlsruhe-Frankfurt: 140km $\Rightarrow$ 1.4ms
- Karlsruhe-Hamburg: 500km $\Rightarrow$ 5ms
- Madrid-Krakow: 2140km $\Rightarrow$ 21ms
- Karlsruhe-New York: 6200km $\Rightarrow$ 62ms

20ms sind deutlich spürbar.[5]

$\Rightarrow$ Cache minimieren + Fehlertolerant codieren. Vorwärtskorrektur (Foreward Error Correction: FEC) gleicht mit Paritätspaketen verlorene Pakete aus.

[5] Auswirkungen von Latenzen beim Tippen (bei xml unsauber: ohne Farben für IntelliJ): https://pavelfatin.com/typing-with-pleasure/

# Latenz Real

```
$ ping uni-hamburg.de
64 bytes from 134.100.36.5: icmp_seq=0 ttl=246 time=40,793 ms
64 bytes from 134.100.36.5: icmp_seq=1 ttl=246 time=81,462 ms
64 bytes from 134.100.36.5: icmp_seq=2 ttl=246 time=40,622 ms
64 bytes from 134.100.36.5: icmp_seq=3 ttl=246 time=40,096 ms
```

```
$ sudo traceroute uni-hamburg.de
  1   192.168.2.1  0,339ms  0,748ms  0,186ms
  2   62.155.245.143  22,924ms  17,829ms  17,195ms
  3   217.0.198.229  22,654ms  22,100ms  22,116ms
  4   217.0.198.229  22,590ms  22,112ms  21,399ms
  5   *  *  *
  6   4.69.142.209  41,394ms  41,031ms  41,059ms
  7   195.122.181.62  42,341ms  39,598ms  40,828ms
  8   188.1.231.82  73,916ms  47,432ms  47,750ms
  9   134.100.254.173  40,370ms  39,352ms  39,368ms
 10   134.100.36.5  40,333ms  39,651ms  39,568ms
```

# Weitere Aspekte

Protokolle für paketbasierte interaktive Kommunikation

- H.323
  - Schnittstelle zwischen Telefonnetz und Internet.
  - RTCP für Steuerung, UDP für Daten.
  - Gateway vermittelt zwischen Telefon und Internet.
- SIP: Session-Initiation-Protocol
  - Liefert Umleitungsserver und externe IP.
  - INVITE, ACK, BYE, CANCEL, REGISTER (Umleitungsserver)

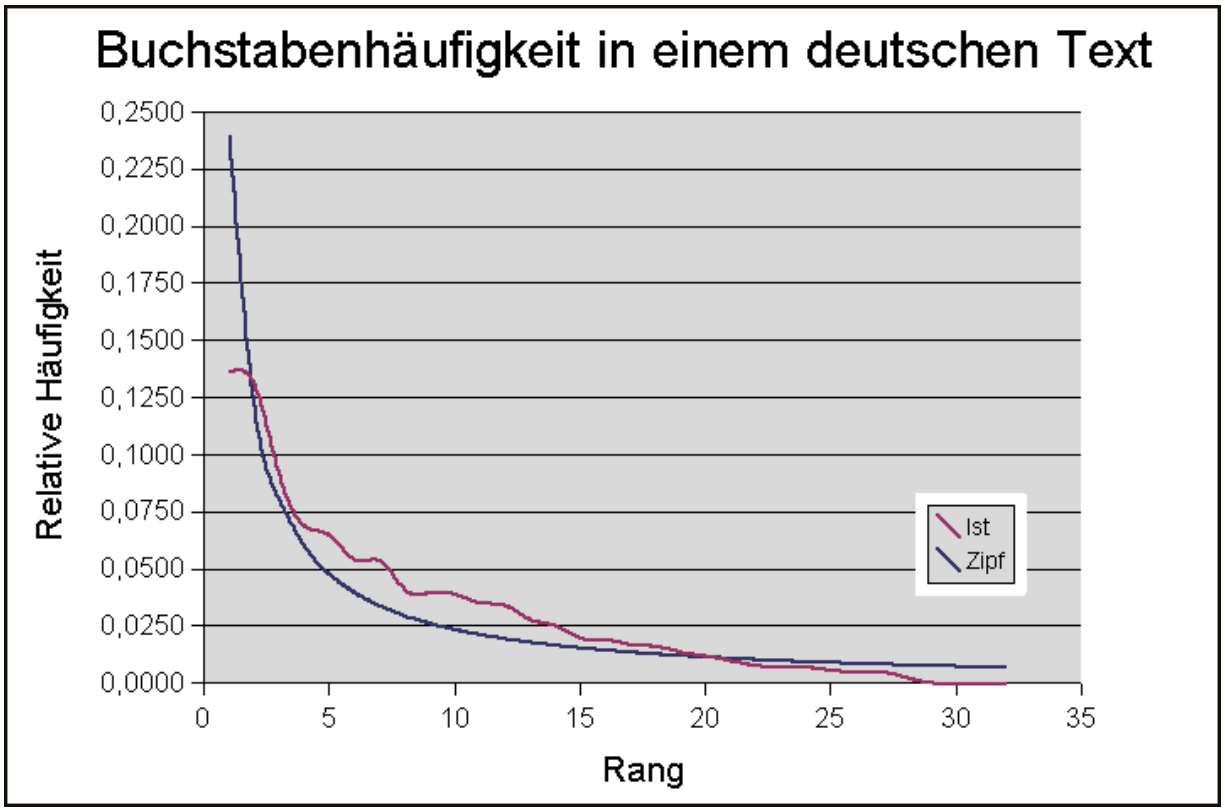
# Streaming Media: Zusammenfassung

- Codecs reduzieren die Bandbreite um Faktor 100
- Video on Demand über TCP
- Interaktiv: RTSP zur Cache-Optimierung
- SIP zum Aufbau von Verbindungen (Sitzungs-Protokoll — OSI-Schicht 5 in Anwendungsschicht)

# Datenverteilung

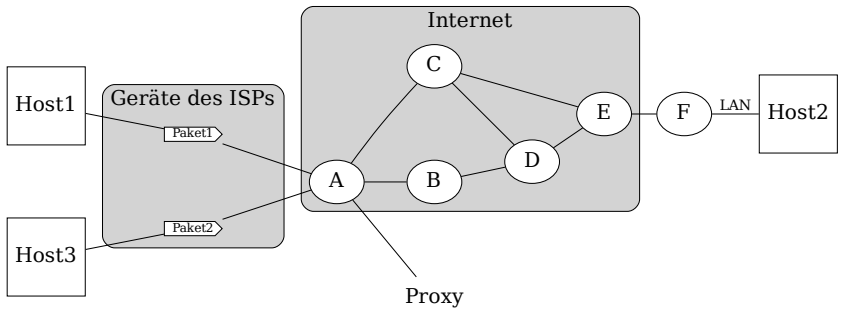
- Häufigkeit der Zugriffe
- Proxy
- Content Delivery Network (CDN)
- peer-to-peer (p2p)

# Häufigkeitsverteilung der Zugriffe: Zipfs Law



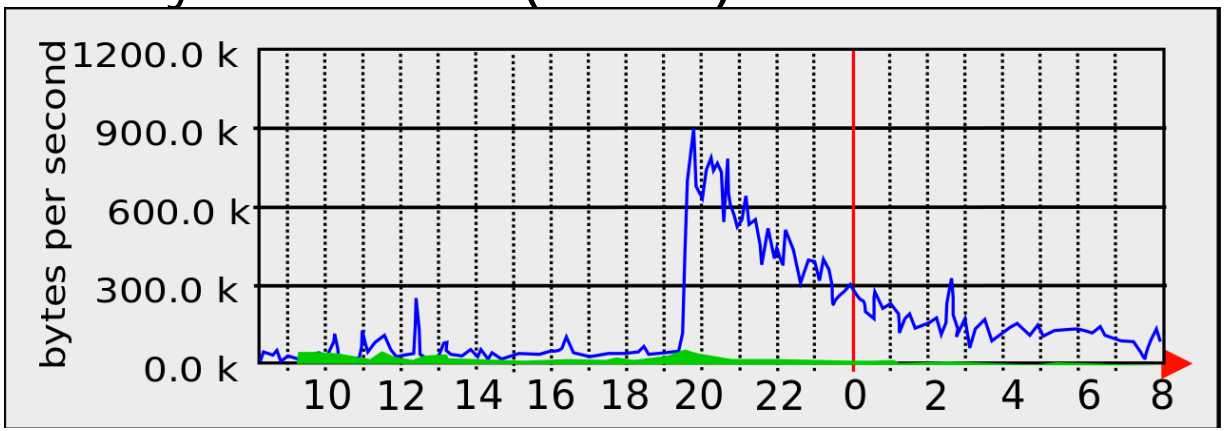

*von Anton: commons.wikimedia.org/wiki/File:Zipf-Verteilung-Buchstaben.png*

# Proxy




- Cache des Internetanbieters (ISP)
- Ruft statische Dateien nur einmal ab $\rightarrow$ spart Bandbreite
- Invalidierung: HTTP-Header für Lebenszeit (z.B. modified-since)
- Kontrolliert vom Internetanbieter

# Content-Delivery-Network (CDN)



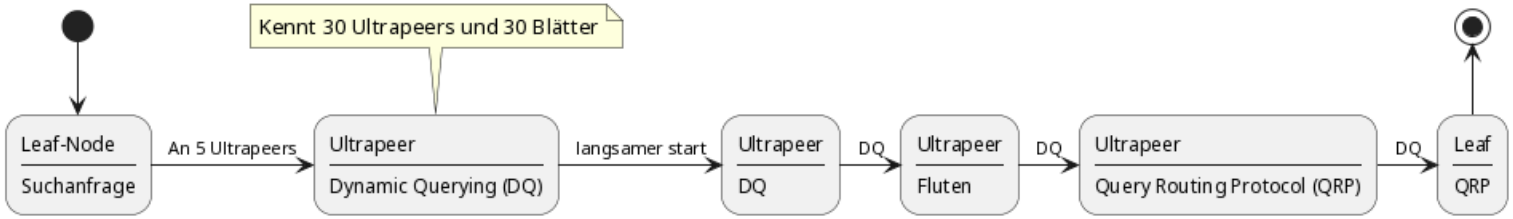

- Regional verteilte Server von Fremdanbietern
- Kontrolliert vom Anbieter. Beispiele: Akamai, Cloudflare, AWS
- Vertrag mit Webseitenbetreiber, Einfluss auf die Webseiten, unterstützt SSL.
- Über DNS zugewiesen
- Vermeidet den Slashdot-Effekt

Original von Fangz, SVG von Teehee123:
commons.wikimedia.org/wiki/File:SlashdotEffectGraph.svg

# peer-to-peer (p2p)

- Nutzer bieten untereinander Dienste an
  - Verteilte Suche
  - Verteilter Index
  - Gemeinsame Inhaltsverteilung (swarming)
- Selbstskalierende Dienste
  - Kapazität steigt mit der Anzahl der Nutzer
  - Kosten steigen üblicherweise nur logarithmisch

# Beispiel für verteilte Suche: Gnutella 0.6 (50 mio Nutzer)




- Dynamic Querying
  - Einen nach dem anderen anfragen (alle binnen etwa 3 Sekunden)
  - bei ausreichend Antworten abbrechen
- Query Routing Protocol
  - Hash-Tabellen mit schwachem Hash auf Suchwort
  - Anfragen erreichen nur Knoten mit wahrscheinlichen Treffern
  - Inter-Ultrapeer QRP: Zusammenfassen der Tabellen.

*weiterlesen:*

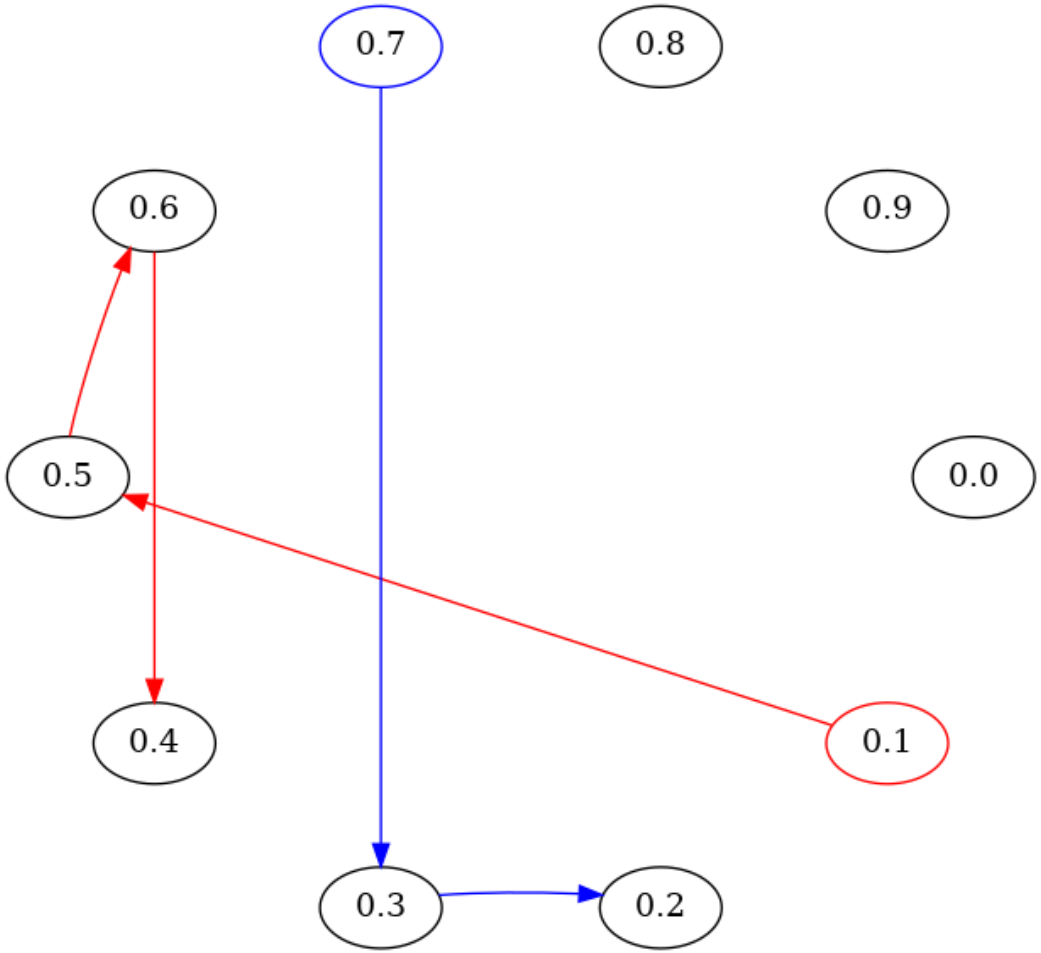
*http://rfc-gnutella.sourceforge.net/src/qrp.html*

# Verteilter Index: Kademlia

- Abstand: (sha1 Knoten-ID) XOR (sha1 Daten) → `1011 XOR 1000 = 0011`
- Knotenlisten (k-Buckets): 160 Listen mit jeweils k Einträgen: IP-Adresse, UDP-Port und Node-ID. Least-recently-seen-queue.
- Bedingung: Jeder Eintrag in Liste n hat die ersten n bits gleich.
- Suche nach Daten: Anfrage an alle in der passenden Liste:
  - Gib mir die Besten Knoten für den Hash X.
  - Die besten Knoten behalten und wieder fragen.
  - Je näher die Knoten am Ziel sind, desto mehr der ihnen bekannte Knoten sind nahe am Ziel.
- Sucht exakten Hash.

*weiterlesen:* *https://en.wikipedia.org/wiki/Kademlia* +

# Verteilter Index: Freenet



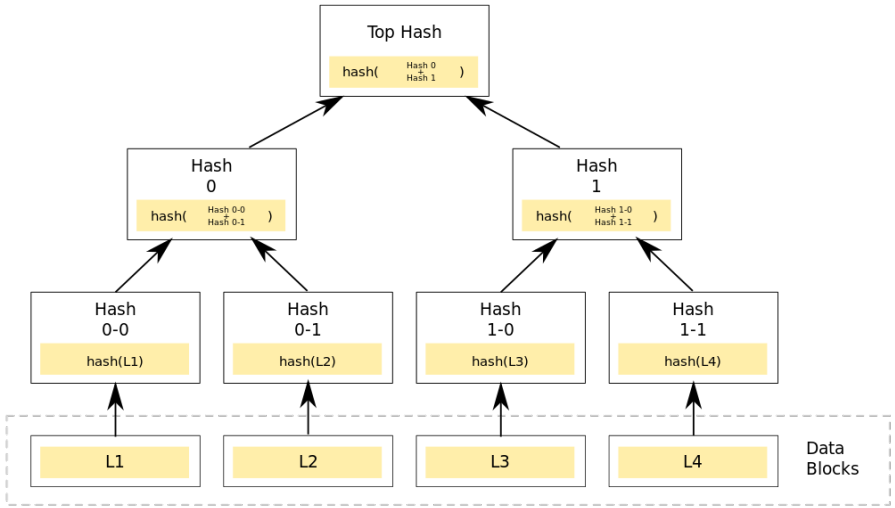
# Gemeinsame Inhaltsverteilung (swarming): Download Mesh

Swarming über 4 zusätzliche HTTP-Header:

- X-Alt: 1.2.3.4:6347,1.2.3.5
- X-NAlts: 1.2.3.4:6346, 1.2.3.5:6341
- X-Gnutella-Content-URN: urn:bitprint:[32-character-SHA1].[39-character-TigerTree]
- Validierung mit Tiger Tree Hash (Merkle-Tree)
  - X-Thex-URI: <URI> ; <ROOT>
- Range-Requests für Dateischnipsel
- Zugriff via Hash: `GET /uri-res/N2R?[URN] HTTP/1.0`

*weiterlesen: http://rfc-gnutella.sourceforge.net/developer/tmp/download-mesh.html (und Links darin: HUGE und PFSP)*

# Download Mesh: Merkle Tree




*Hash Tree von Azaghal*
*commons.wikimedia.org/wiki/File:Hash_Tree.svg*

*Torrents zentralisieren swarming auf Tracker-Server mit Statistiken und Community.*

# Zusammenfassung

Proxy:

- Vom Internetanbieter betrieben
- Zwischenspeicher für Daten
- Spart Bandbreite

CDN:

- Dienstleistung für Webseiten
- Bei Internetanbietern aufgestellt, aber von CDN-Betreiber Kontrolliert

Peer-to-Peer:

- Selbstskalierend
- Fuzzy-Suche: Gnutella
- Hash-Suche: Kademlia
- Swarming: Dateien aus vielen Quellen

# Rückmeldung

- Was sollte ich beibehalten?
- Was sollte ich ändern?

# Zusammenfassung

- E-Mail ist ein verteiltes Protokoll zwischen verschiedenen Anbietern
  - E-Mails werden mit SMTP weitergeleitet
  - Nicht-Text-Inhalte können in MIME gekapselt werden
- Streaming verwendet hoch-optimierte Codecs
  - Video-on-Demand kann mehr Optimierungen nutzen als interaktive Konferenzen
- Mittel zur Datenverteilung sind
  - Proxies (beim Internetanbieter)
  - CDNs (vom Webseitenbetreiber)
  - Peer-to-Peer-Netze (bei den Nutzern)

# Fragen für die Prüfung?

Ideensammlung:

- 36 GiB für 1h 720p Video, kann das ein moderner Codec sein?
- Nennen Sie 2 Beispiele für Infomationen, die in den Envelope einer E-Mail gehören.
- Beschreiben Sie den Unterschied zwischen Envelope und Header

# Selbststudium diese Woche I

- Schreiben Sie einen Webserver, der auf die ersten drei SMTP-Anfragen antworten kann. Verbinden Sie sich mit telnet mit Ihrem Server und dokumentieren Sie die Interaktion. Sie brauchen **kein** STARTTLS zu implementieren. Wählen Sie dafür entweder die Sprache, in der Ihnen die Aufgabe **leichter** fällt. Die Sprachen erhalten Sie wieder von dem Sprachpaargenerator (sie sind noch gleich): https://www.draketo.de/software/vorlesung-netztechnik#nummer-zu-sprache (läuft clientseitig in Ihrem Browser).
- Zeigen Sie, wie weit sie mit TELNET mit dem wirklichen SMTP-Server ihres E-Mail-Providers sprechen können.

# Als nächstes: Werkzeuge für eigene Anwendungen

# Verweise I

Bilder: