HTTP - Hypertext Transfer Protocol

© Martina Kannen

Inhalt

1. HTTP im Überblick
2. Kommunikationsmodell
3. Laborübung
4. HTTP 2.0/ SPDY
5. Laborübung
6. QUIC

© Martina Kannen

HTTP im Überblick

1991: Tim Berners-Lee (CERN) entwickelte HTTP

1991: HTTP 0.9, W3C, Übertragung von HTML-Dokumente vom Server zum Client

1992: HTTP 1.0, Übertragung beliebiger Dateiformate zwischen Client und Server.

1996: RFC 1945, HTTP 1.0

1997: HTTP 1.1 (RFC 2068), 1999: aktueller RFC 2616

2009: Erste SPDY-Version

2015: HTTP 2.0, RFC 7540

2017: HTTP over QUIC, Draft

2017: QUIC: An UDP-based Multiplexed and Secure Transport, Draft

© Martina Kannen

HTTP im Überblick

Designziele

1. **Einfachheit: Das Protokoll soll nur wenige Ressourcen beanspruchen.**

2. **Geschwindigkeit: Das Protokoll soll so schnell wie möglich sein.**

© Martina Kannen

HTTP im Überblick

Designziel - Geschwindigkeit

- **Schnellere Webseiten haben eine geringere Bounce- und eine höhere Konversionsrate**
- **Faustformel: 1 Sekunde langsamer bedeutet 5 % geringere Konversionsrate**
- **Schnellere Webseiten haben ein besseres Google-Ranking**

<u>Bouncerate:</u> Absprungsrate (nur einzelne Seiten werden besucht oder Kurzbesuche von 5-7 Sekunden)

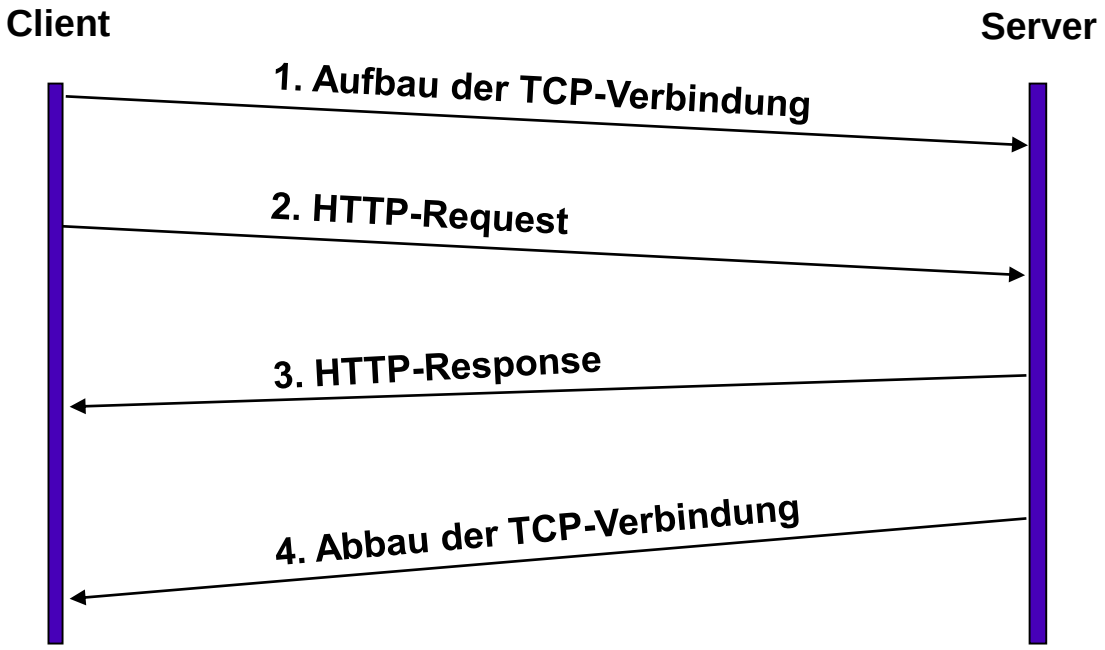
<u>Konversionsrate:</u> Wandlung vom Interessenten zum Käufer

Wintermeyer, Stefan: Schneller Wechsel, IX 8, 2015

© Martina Kannen

Kommunikationsmodell

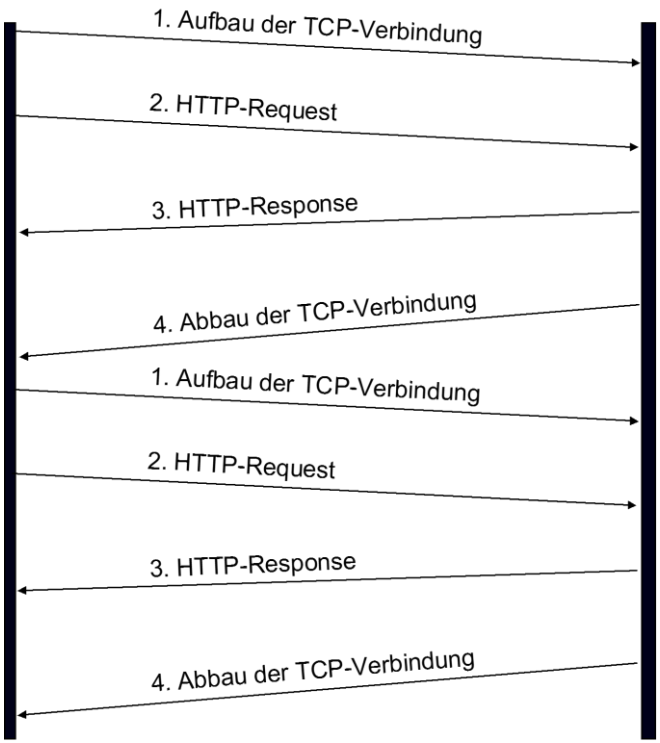
Kommunikationsablauf HTTP 1.0



© Martina Kannen

Kommunikationsmodell

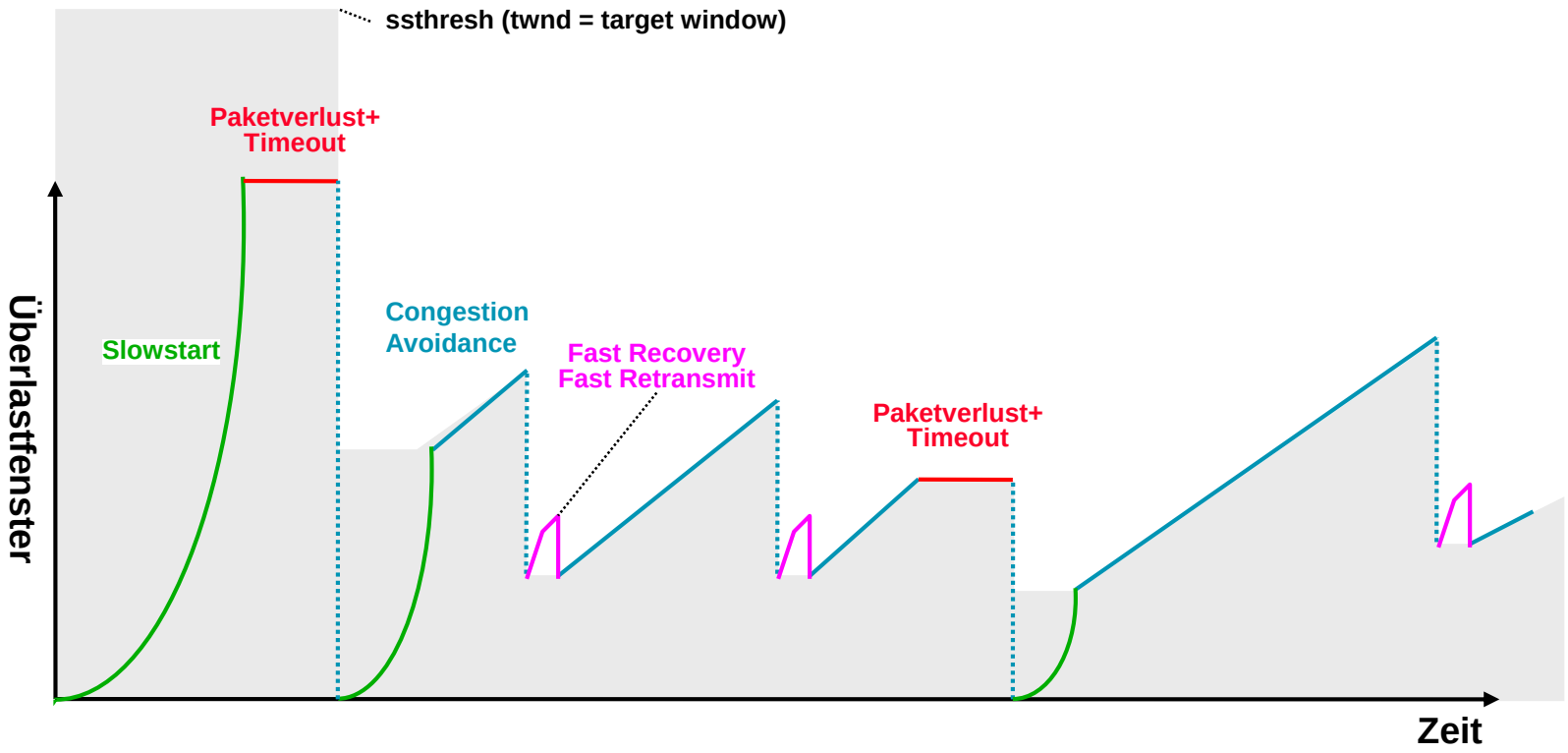
Kommunikationsablauf HTTP 1.0



© Martina Kannen

Kommunikationsmodell

TCP-Sägezahnverhalten



Leischner, Martin: Vorlesung Internetkommunikation WS 10/11.

© Martina Kannen

Kommunikationsmodell

Mögliche Lösungsansätze

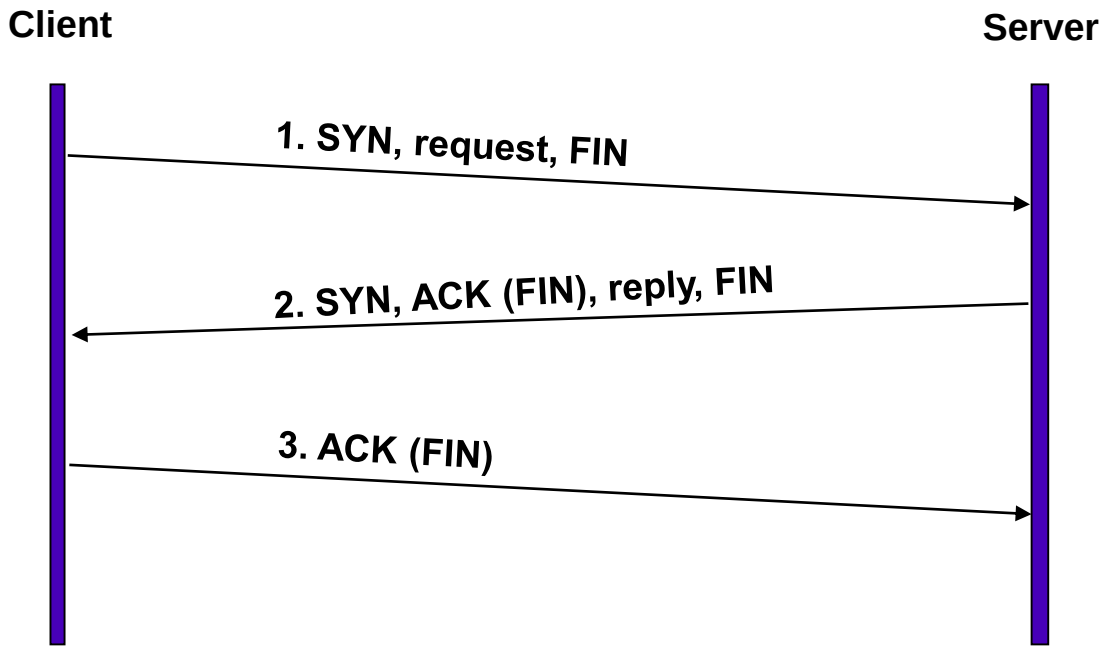
- **HTTP over multiple, concurrent TCP connections**
- **Persistent-HTTP (P-HTTP)**
- **HTTP over UDP-based protocols**
- **Transaction-TCP (T/TCP)**
- **Neues Transportprotokoll**

Heidemann, John, Obraczka, Katia, Touch, Joe: Modeling the Performance of HTTP Over Several Transport Protocols. IEEE/ ACM Transactions on Networking, 5(5), 1997.

© Martina Kannen

Kommunikationsmodell

Transaction-TCP

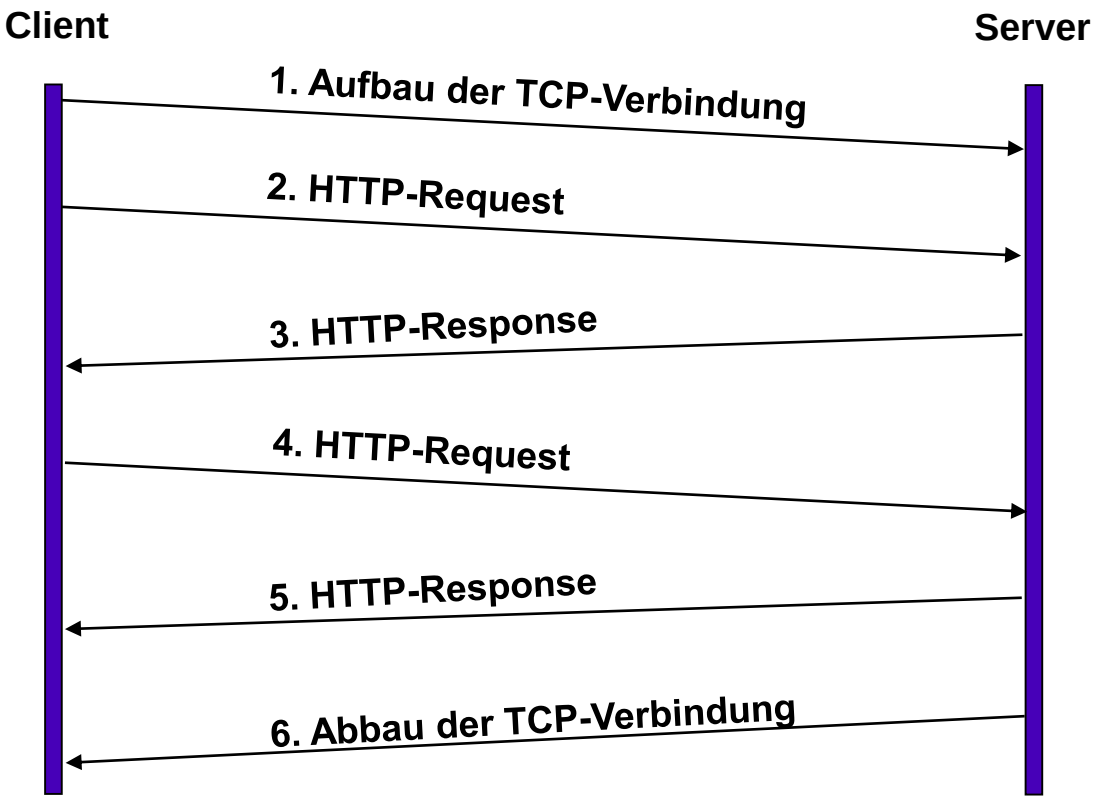


Tanenbaum, Andrew: Computernetzwerke. Pearson Studium, 2003, nach RFC 1378 und 1644

© Martina Kannen

Kommunikationsmodell

Persistent-HTTP 1.1

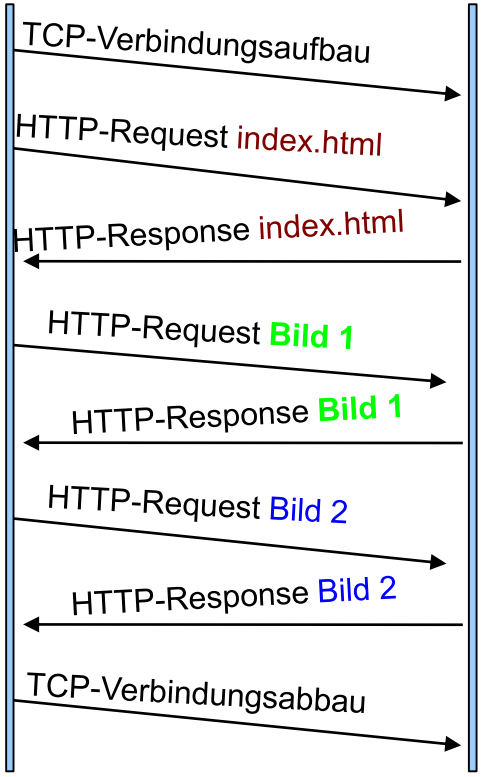


© Martina Kannen

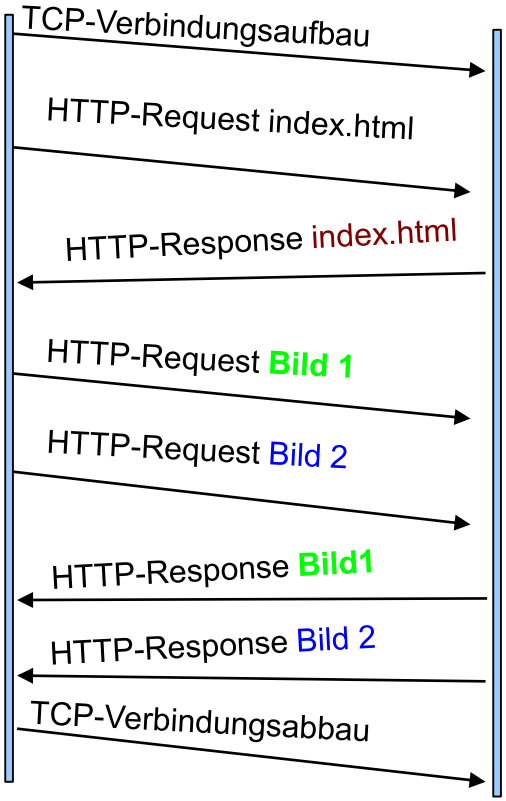
Kommunikationsmodell

Persistent-HTTP 1.1 ohne und mit Pipelining

HTTP 1.1 ohne Pipeling



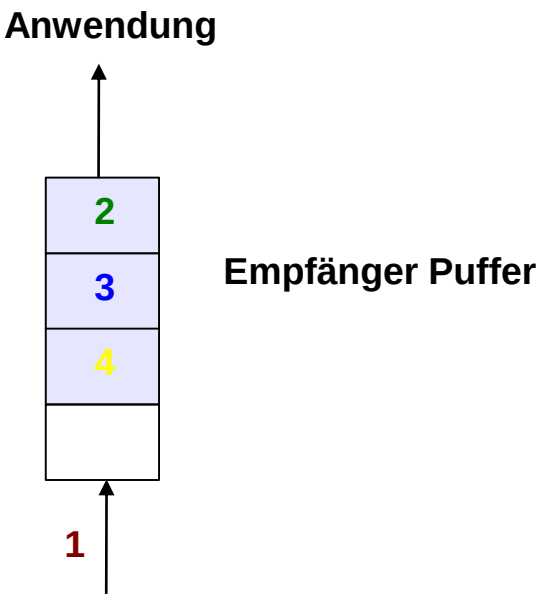
HTTP 1.1 Pipelining



© Martina Kannen

Kommunikationsmodell

Head-of-Line (HOL) Blocking



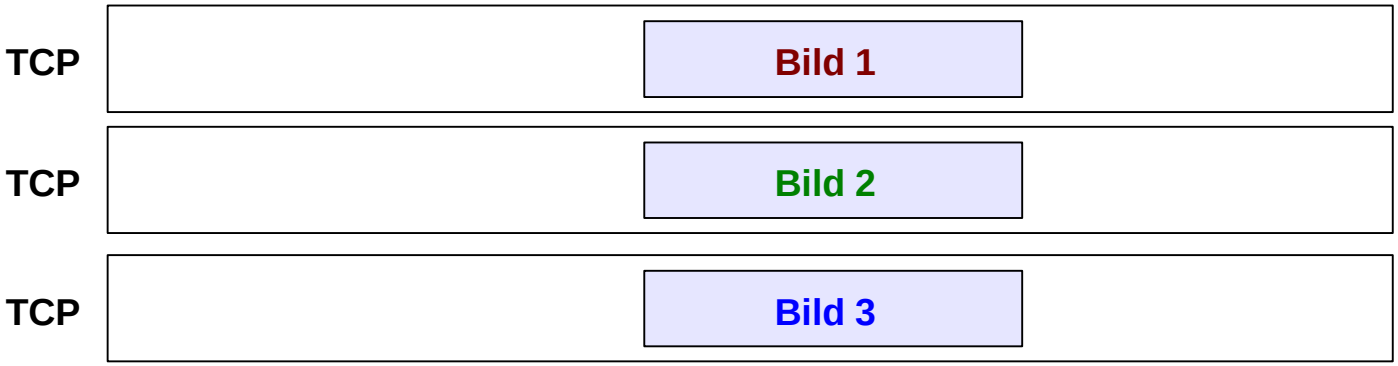
Kang, Sukwoo, Fields Matthew: Experimental Study of the SCTP compared to TCP, Computer Communications and Networking Fall, Project Report, Electrical Engineering Department, 2003.

© Martina Kannen

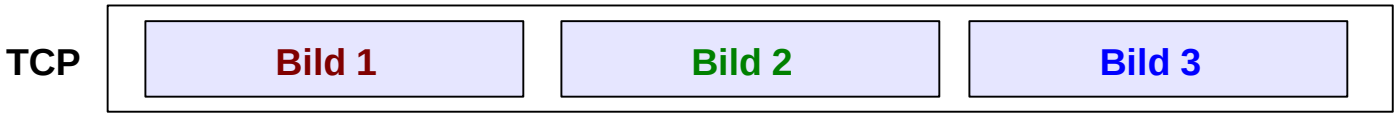
Kommunikationsmodell

Head-of-Line Blocking

- HTTP 1.0 over TCP: Teure Ressourcen, kein HOL-Blocking



- HTTP 1.1 over TCP: Sparsam mit Ressourcen, HOL-Blocking



Becker, Markus: Stream Control Transmission Protocol -Einsatz in der Praxis anhand eines Webservers, Abschlussarbeit BCS Hochschule Bonn-Rhein-Sieg, Fachbereich Informatik, September 2009.

© Martina Kannen

Kommunikationsmodell

Protokolleigenschaften von HTTP 1.1

- Basiert auf TCP
- Persistente Verbindungen
- Request-/ Response-Abfolge (ohne Pipelining)
- Client-Initialisierung
- Bidirektionaler Datentransfer
- Zustandslos
- Nicht-sitzungsorientiert

© Martina Kannen

Kommunikationsmodell

Performance-Betrachtung

	HTTP 1.0	HTTP 1.1 Persistent	HTTP Pipelining
Max. gleichzeitige Verbindungen	6	1	1
Anzahl verwendeter Sockets	40	1	1
Pakete vom Client zum Server	226	70	25
Pakete vom Server zum Client	271	153	58
Pakete insgesamt	497	223	83
Zeit insgesamt (in Sekunden)	1,85	4,13	3,02

Nielsen, Hendrik Frystyk, Gettys, Jim, Baird-Smith, Anselm, Prud'hommeaux, Eric, Håkon Wium Lie, Lilley, Chris: Network Performance Effects of HTTP/1.1, CSS1, and PNG, W3C, NOTE 24-June 1997.

© Martina Kannen

Kommunikationsmodell

Performance-Betrachtung ohne Nagle-Algorithmus

	HTTP 1.0	HTTP 1.1 Persistent	HTTP Pipelining
Pakete insgesamt	510,2	281	181,8
Transfervolumen in Bytes	216.289	191.843	191.551
Zeit insgesamt (in Sekunden)	0,97	1,25	0,68

Nielsen, Hendrik Frystyk, Gettys, Jim, Baird-Smith, Anselm, Prud'hommeaux, Eric, Håkon Wium Lie, Lilley, Chris: Network Performance Effects of HTTP/1.1, CSS1, and PNG, W3C, NOTE 24-June 1997.

© Martina Kannen

Kommunikationsmodell

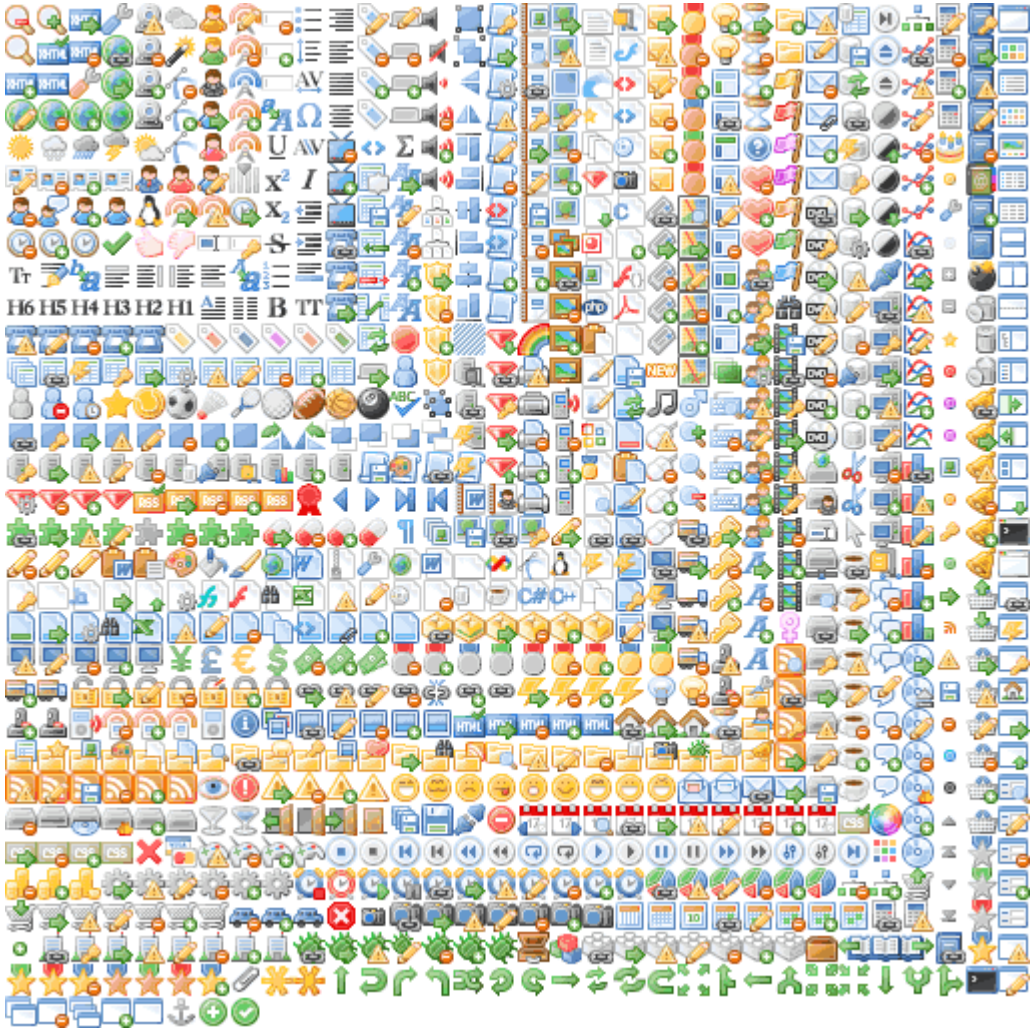
Optimization Hacks/ weitere Performanceoptimierungen

- **Seiten-Tuning: z.B. Ajax, CSS-Sprites, Domain-Sharding**
- **Client-seitig: z.B. Caching**
- **Server-seitig: z.B. Load-Balancing, Content Delivery Networks (CDNs)**
- **Transportprotokoll: Neue Transportprotokolle: z.B. SCTP, TCP-Optimierung: z.B. MPTCP, UDP: z.B. Google QUIC-Protokoll**
- **Neue HTTP-Version**

© Martina Kannen

Kommunikationsmodell

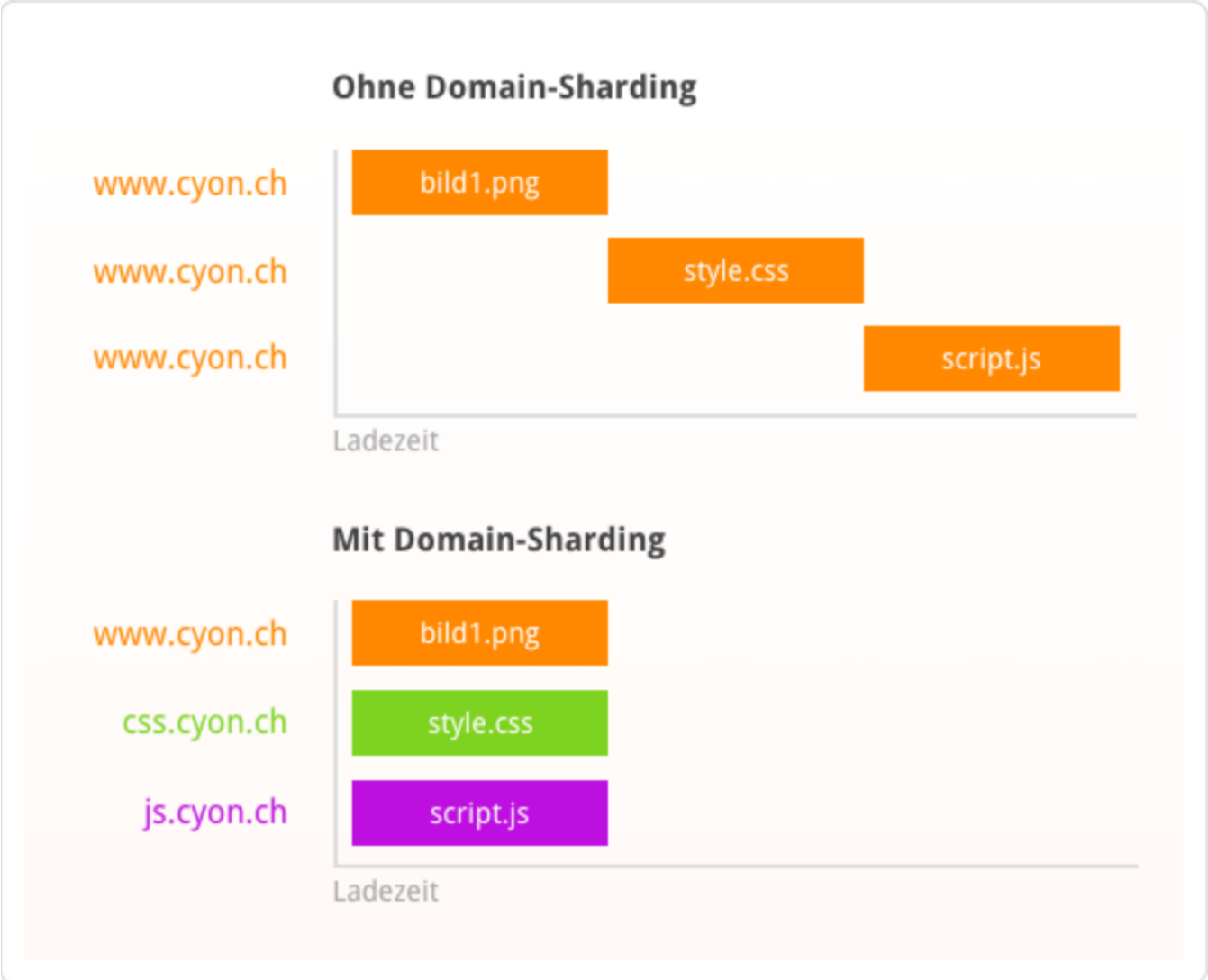
CSS-Sprites



© Martina Kannen

Kommunikationsmodell

Domain Sharding



© Martina Kannen

Philipp Zeder: Domain-Sharding: Eine Technik die Sie 2016 vergessen können, https://www.cyon.ch/blog/Domain-Sharding, 8. Januar 2016

HTTP 2.0

- **Google-Projekt: Let´s make the web faster, Mike Belshe und Roberto Peon**
- **1. SPDY-Version Ende 2009, letzte Version 2016: SPDY 3**
- **Ziel: Reduzierung der Webseiten-Latenz für User; lt. Google Lab: 55 % geringere Page Load Time mit SPDY**
- **Mai 2015: HTTP 2.0 RFC 7540**
- **Browser: Firefox, Chrom, Opera, IE/Edge (11, ab Windows 10), Safari, Amazon Silk**
- **Server: Apache, Caddy, Jetty, nginx, node.js**
- **CDN: Akamai, Cloudflare, AWS, Fastly…**
- **Websites: Google, Instagram, Wikipedia, Twitter…**

© Martina Kannen

HTTP 2.0

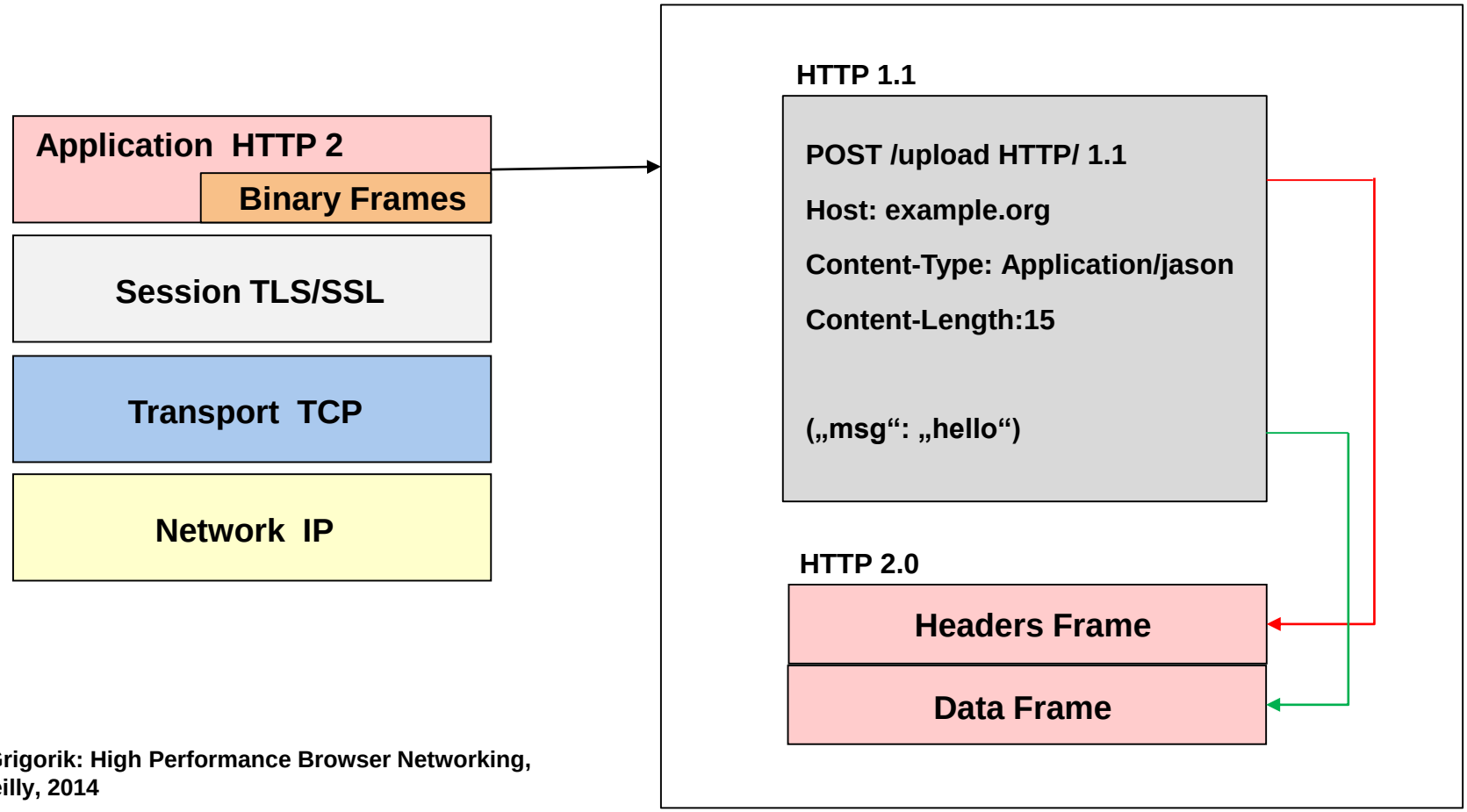
Merkmale

- **Binäres Format**
- **Multiplexing**
- **Stream-Priorisierung**
- **Server-Push**
- **HTTP-Header-Kompression: RFC 7541 (HPACK)**
- **SSL/TLS (SPDY “must”, HTTP 2.0 “optional”)**

© Martina Kannen

HTTP – 2.0

Protokollarchitektur

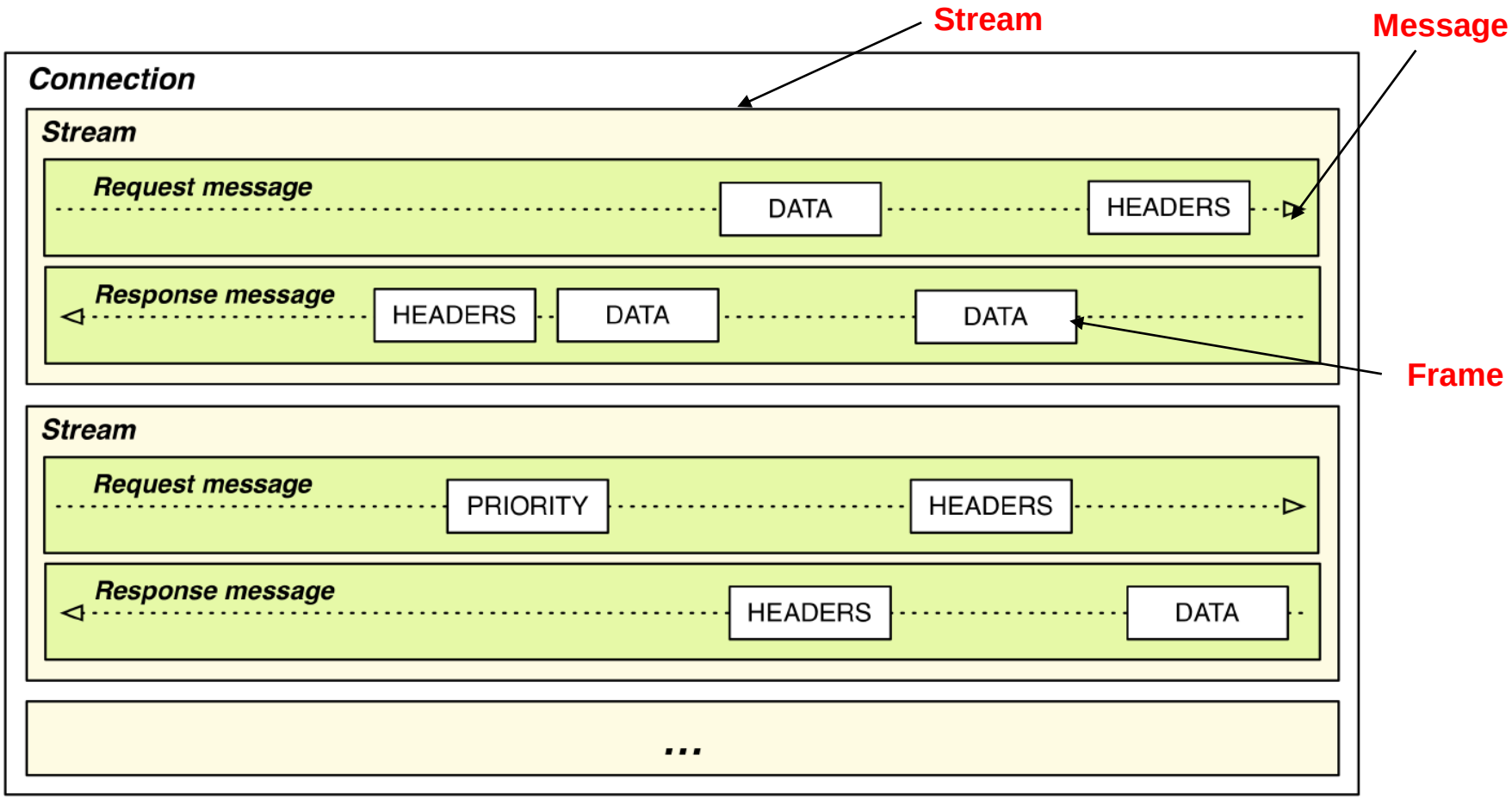


Ilya Grigorik: High Performance Browser Networking, O´ Reilly, 2014

© Martina Kannen

HTTP 2.0

Neue Terminologie: Streams, Messages und Frames



Ilya Grigorik: High Performance Browser Networking, O´ Reilly, 2014

© Martina Kannen

HTTP 2.0

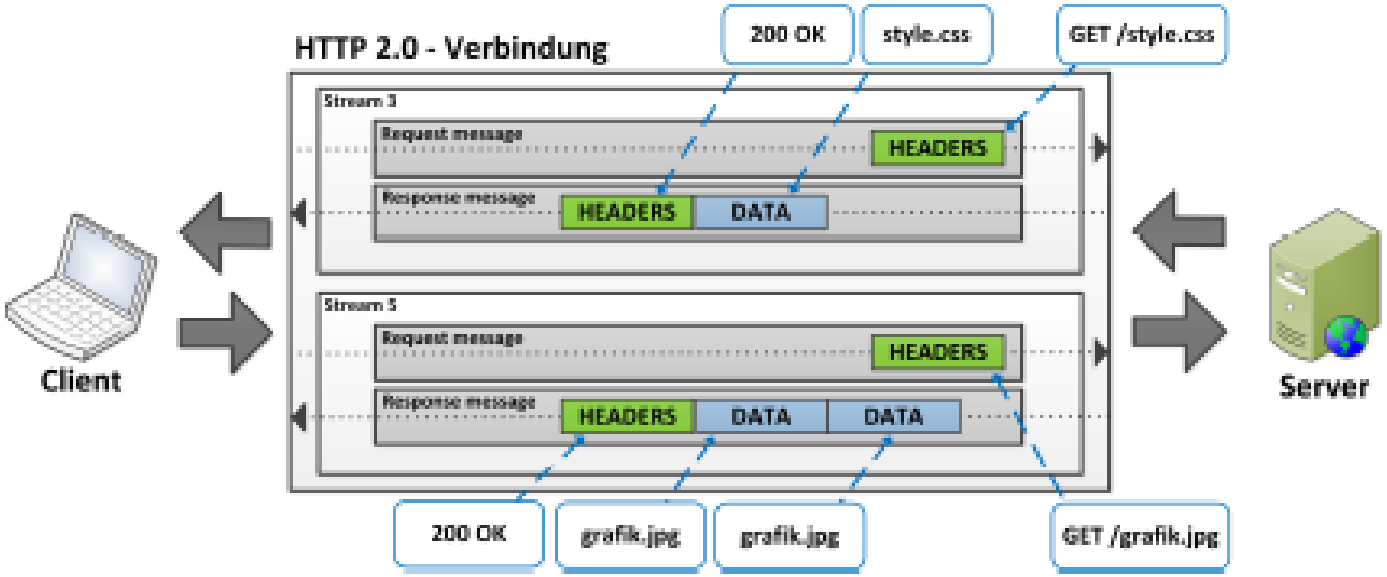
Streams, Messages und Frames

- **Jede HTTP-Kommunikation zwischen einem Client und einem Server wird mit einer TCP-Verbindung durchgeführt.**
- **Ein Stream ist ein virtueller Kanal in einer Verbindung, der bidirektionale Messages transportiert.**
- **Eine Message ist eine logische HTTP-Nachricht (Request oder Response), die eine oder mehrere Frames enthält**
- **Frames sind die kleinste Einheit der Kommunikation, die spezifische Typen von Daten wie Header oder Payload transportieren.**

© Martina Kannen

Kommunikationsmodell

Streams, Messages und Frames



Ilya Grigorik: High Performance Browser Networking, O´ Reilly, 2014

© Martina Kannen

HTTP 2.0

Streams

"A "stream" is an independent, bidirectional sequence of frames exchanged between the client and server within an HTTP/2 connection."

- **Eine Verbindung kann mehrere offene Streams enthalten.**
- **Streams können von jedem Endpunkt geöffnet und geschlossen werden.**
- **Streams können uni- oder bidirektional sein.**
- **Strikte Einhaltung der Frame-Reihenfolge innerhalb eines Streams.**
- **Jeder Stream besitzt einen Stream-Identifier (Client ungerade/ Server gerade Nummern).**

Belshe M. (von BitGo), Peon R. (v. Google), M. Thomson, Ed. (v. Mozilla): Hypertext Transfer Protocol Version 2 (HTTP/2), RFC 7540, Mai 2015.

© Martina Kannen

HTTP 2.0

Frames

```
+-----------------------------------------------+
|                 Length (24)                   |
+---------------+---------------+---------------+
|   Type (8)    |   Flags (8)   |
+-+-------------+---------------+-------------------------------+
|R|                 Stream Identifier (31)                      |
+=+=============================================================+
|                   Frame Payload (0...)                      ...
+---------------------------------------------------------------+
```

- **Jeder Frame besteht aus einem Header und einer Payload**
- **Der Stream Identifier zeigt den zugehörigen Stream an**
- **Frame-Typen: DATA, HEADERS, PRIORITY, RST_STREAM, SETTINGS, PUSH-PROMISE, PING, GOAWAY, WINDOW_UPDATE, CONTINUATION**

Belshe M., Peon R., M. Thomson, Ed.: Hypertext Transfer Protocol Version 2 (HTTP/2), RFC 7540, Mai 2015.

© Martina Kannen

HTTP 2.0

Frame-Typen

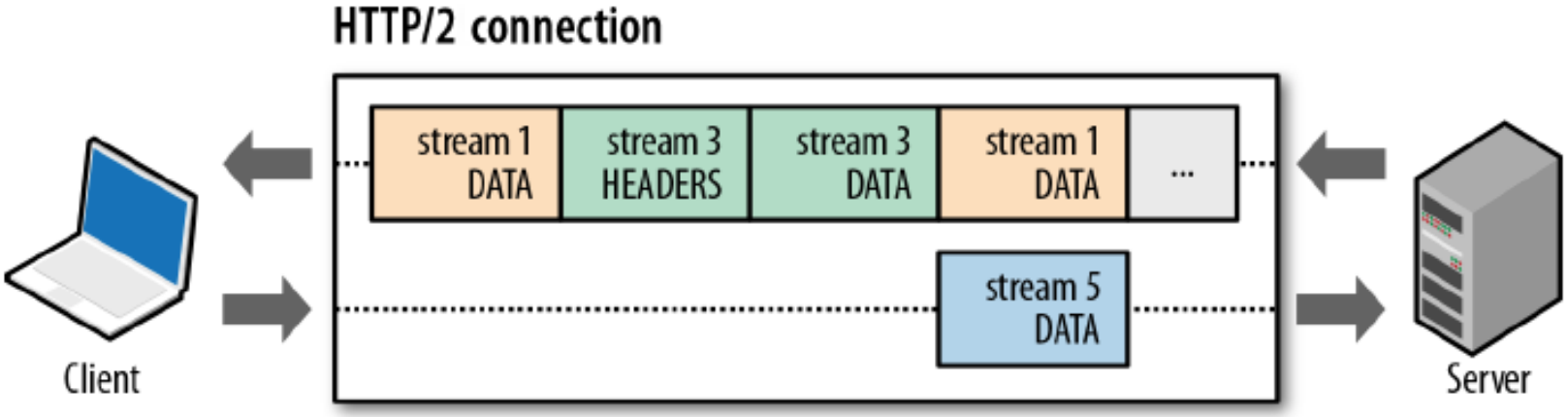
Frame-Typ	Bedeutung
DATA	HTTP-Message-Bodies (Payload)
HEADERS	Öffnen eines Streams + HTTP-Message-Header
PRIORITY	Priorisierung eines Streams
RST_STREAM	Direkte Unterbrechung eines Streams
SETTINGS	Konfigurationsparameter pro Verbindung
PUSH_PROMISE	Vorschlag/Information für Server-Push
PING	RTT-Messung + Liveness-Check
GOAWAY	Schließen eines Streams
WINDOW_UPDATE	Flow-Control pro Stream/Verbindung
CONTINUATION	Aufrechterhaltung eines Streams

Belshe M., Peon R., M. Thomson, Ed.: Hypertext Transfer Protocol Version 2 (HTTP/2), RFC 7540, Mai 2015.

© Martina Kannen

Kommunikationsmodell

Multiplexing

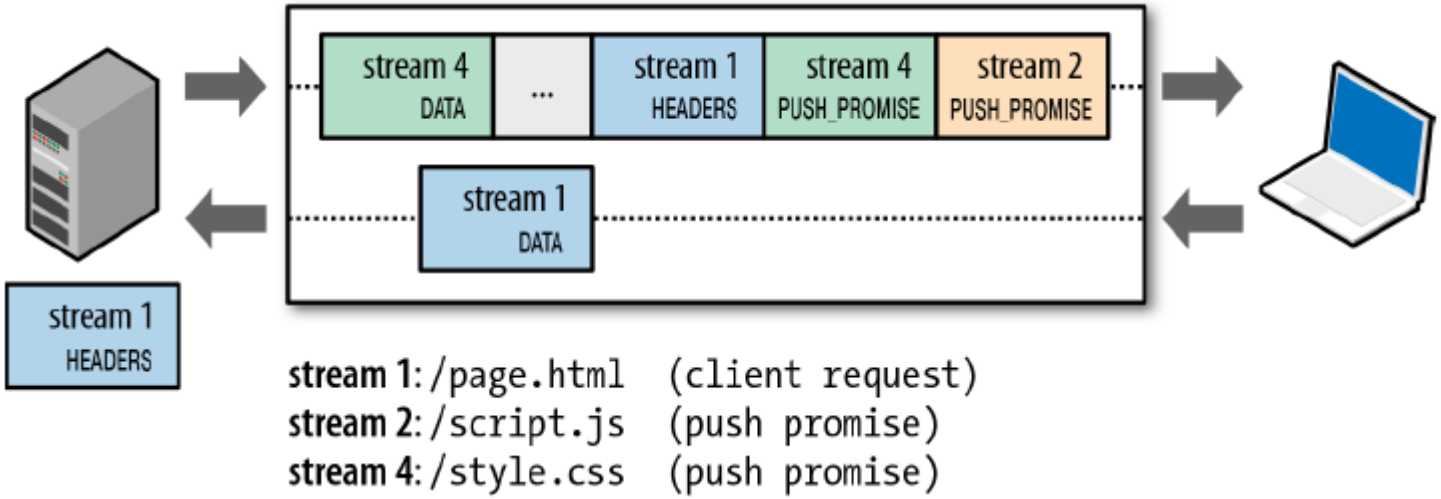


Ilya Grigorik: High Performance Browser Networking, O´ Reilly, 2014

© Martina Kannen

Kommunikationsmodell

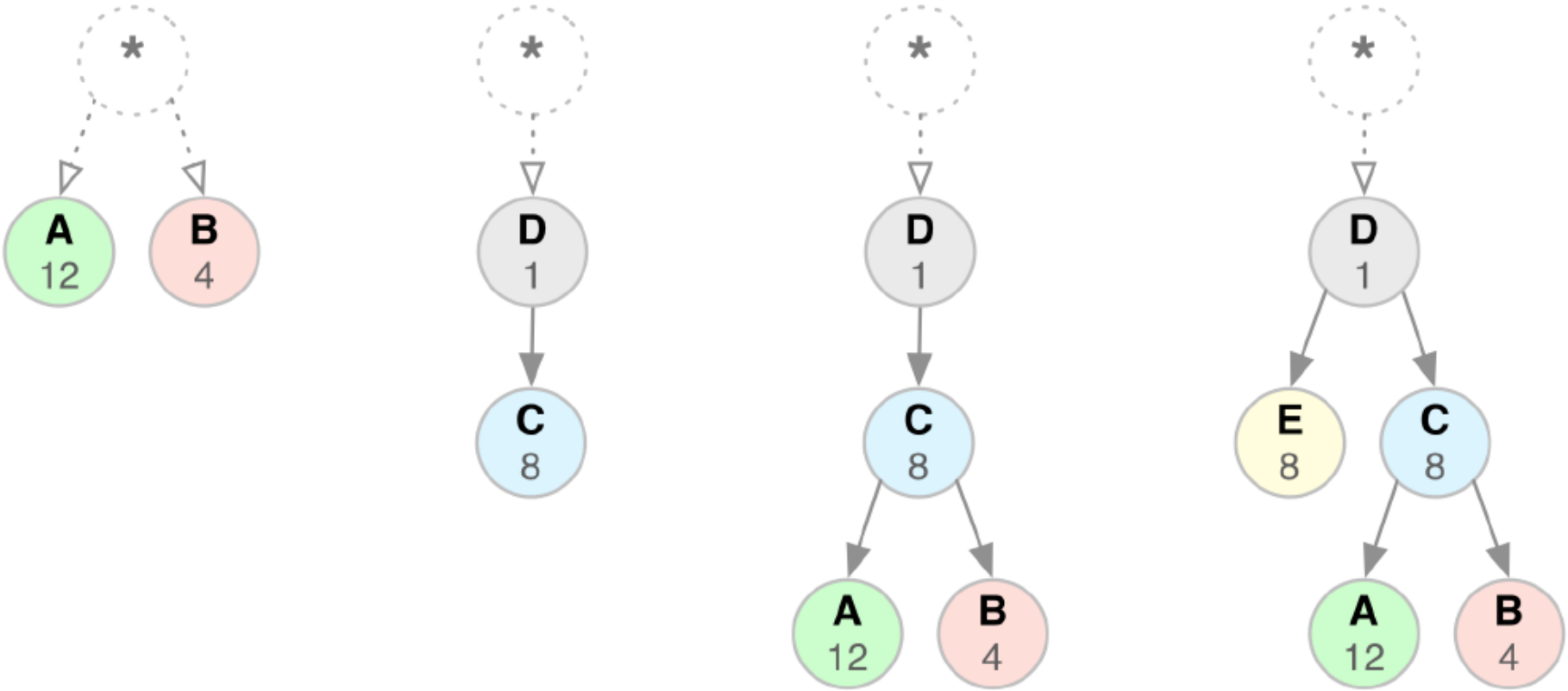
Server Push HTTP 2.0



Ilya Grigorik: High Performance Browser Networking, O´ Reilly, 2014

HTTP 2.0

Priorisierung von Streams: Abhängigkeit und Gewichtung

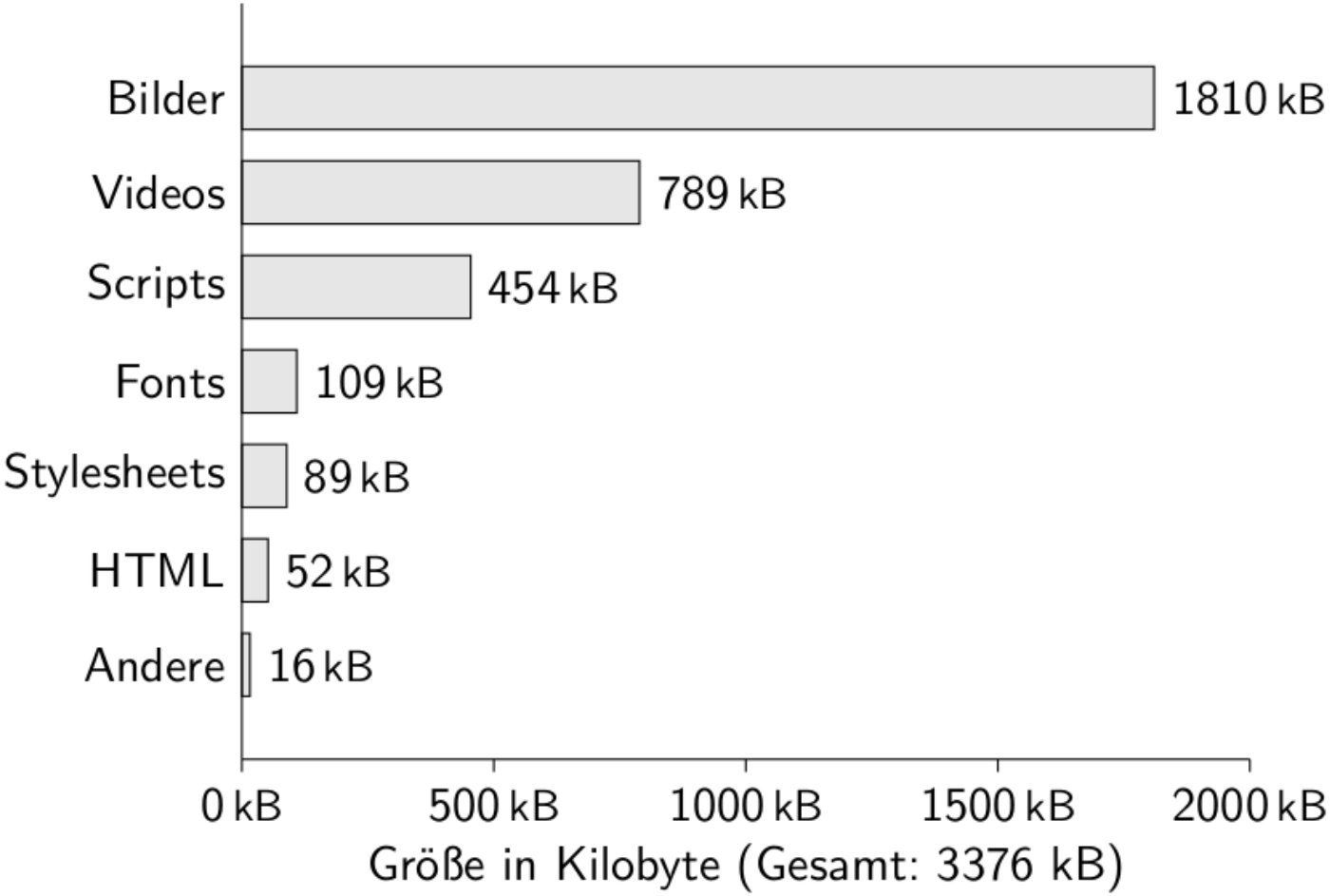


Ilya Grigorik: High Performance Browser Networking, O´ Reilly, 2014

© Martina Kannen

HTTP 2.0

Gemittelte Größe Content-Types pro Webseite nach
`httparchive.org`



Hachtkemper, Manuel: Analyse ausgewählter Aspekte von HTTP/2 Client-Implementierungen. Master Thesis, 2018

© Martina Kannen

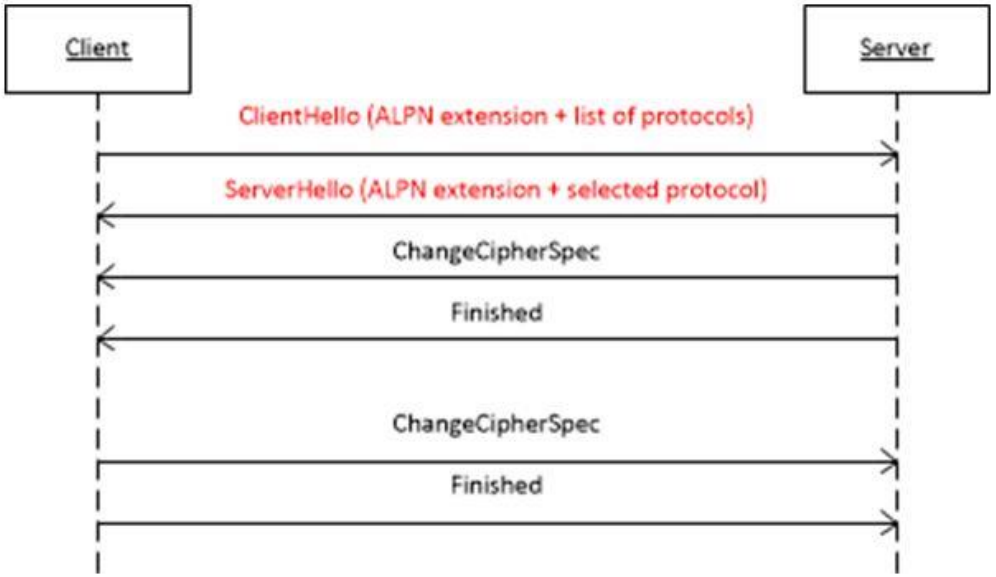
HTTP 2.0

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>the title</title>
        <link rel="prefetch" href="/content/prefetch.html">
        <link rel="import" href="/content/import.html">
        <script src="/content/head-async.js" async></script>
        <script src="/content/head-defer.js" defer></script>
        <script src="/content/head.js"></script>
        <link rel="stylesheet" type="text/css" href="/content/layout.css">
    </head>

    <body>
        <p style="font-family:AileronBlack">HTTP/2 Testwebseite</p>
        <img src="/content/bild.jpg" alt="Bild" width="540" height="360" /><br />
        <img src="/content/bild.png" alt="Bild" width="200" height="141" />
        <script src="/content/body-async.js" async></script>
        <script src="/content/body-defer.js" defer></script>
        <script src="/content/body.js"></script>
        <script>reportEvent('beacon');</script>
        <div id="xhr"> xhr content </div><script>var xhr = new XMLHttpRequest();xhr.onreadystatechange = function() {if (xhr.readyState == XMLHttpRequest.DONE) {document.getElementById
        ("xhr").innerHTML = xhr.responseText;}};xhr.open("GET", "/content/xhr.html", true);xhr.send();</script>
        <iframe src="/content/iframe.html" height="100" width="200"></iframe>
    </body>
</html>
```

HTTP 2.0

Abbreviated TLS Handshake with ALPN



© Martina Rahmer

Hachtkemper, Manuel: Analyse ausgewählter Aspekte von HTTP/2 Master Thesis, 2018

*eth0

Datei Bearbeiten Ansicht Navigation Aufzeichnen Analyse Statistiken Telephonie Wireless Tools Hilfe

tcp.stream eq 2 Ausdruck... +

No.	Time	Source	Destination	Protocol	Length	Info
28	1.256961786	2001:638:408:100:699f…	2001:638:408:100::111	TLSv1.2	179	Client Key Exchange, Change Cipher Spec, Finished
29	1.258319520	2001:638:408:100::111	2001:638:408:100:699f:38ac:e02:5fe3	TLSv1.2	376	New Session Ticket, Change Cipher Spec, Finished
30	1.258904000	2001:638:408:100::111	2001:638:408:100:699f:38ac:e02:5fe3	HTTP2	143	SETTINGS, WINDOW_UPDATE
42	1.274355320	2001:638:408:100::111	2001:638:408:100:699f:38ac:e02:5fe3	TCP	143	[TCP Retransmission] 443 → 44342 [PSH, ACK] Seq=2688430510 Ack=30198…
43	1.274364818	2001:638:408:100:699f…	2001:638:408:100::111	TCP	98	44342 → 443 [ACK] Seq=3019850193 Ack=2688430567 Win=42752 Len=0 TSva…
55	1.299110479	2001:638:408:100:699f…	2001:638:408:100::111	HTTP2	263	Magic, SETTINGS, WINDOW_UPDATE, PRIORITY, PRIORITY, PRIORITY, PRIORI…
56	1.299363993	2001:638:408:100:699f…	2001:638:408:100::111	HTTP2	334	HEADERS, WINDOW_UPDATE
57	1.299679524	2001:638:408:100::111	2001:638:408:100:699f:38ac:e02:5fe3	TCP	86	443 → 44342 [ACK] Seq=2688430567 Ack=3019850618 Win=31872 Len=0 TSva…
58	1.299689651	2001:638:408:100:699f…	2001:638:408:100::111	HTTP2	124	SETTINGS
59	1.300656637	2001:638:408:100::111	2001:638:408:100:699f:38ac:e02:5fe3	HTTP2	124	SETTINGS
60	1.300665297	2001:638:408:100:699f…	2001:638:408:100::111	TCP	86	44342 → 443 [ACK] Seq=3019850656 Ack=2688430605 Win=42752 Len=0 TSva…
61	1.303392777	2001:638:408:100::111	2001:638:408:100:699f:38ac:e02:5fe3	HTTP2	818	HEADERS, DATA
62	1.303399901	2001:638:408:100:699f…	2001:638:408:100::111	TCP	86	44342 → 443 [ACK] Seq=3019850656 Ack=2688431337 Win=45568 Len=0 TSva…
74	1.344427124	2001:638:408:100:699f…	2001:638:408:100::111	HTTP2	200	HEADERS, WINDOW_UPDATE

```
▸ Frame 55: 263 bytes on wire (2104 bits), 263 bytes captured (2104 bits) on interface 0
▸ Ethernet II, Src: 00:1c:c0:a3:f3:d2, Dst: 00:50:56:90:64:7f
▸ Internet Protocol Version 6, Src: 2001:638:408:100:699f:38ac:e02:5fe3, Dst: 2001:638:408:100::111
▸ Transmission Control Protocol, Src Port: 44342, Dst Port: 443, Seq: 3019850193, Ack: 2688430567, Len: 177
▸ Secure Sockets Layer
▾ HyperText Transfer Protocol 2
  ▾ Stream: Magic
      Magic: PRI * HTTP/2.0\r\n\r\nSM\r\n\r\n
  ▾ Stream: SETTINGS, Stream ID: 0, Length 18
      Length: 18
      Type: SETTINGS (4)
    ▸ Flags: 0x00
      0... .... .... .... .... .... .... .... = Reserved: 0x0
      .000 0000 0000 0000 0000 0000 0000 0000 = Stream Identifier: 0
    ▸ Settings - Header table size : 65536
    ▸ Settings - Initial Windows size : 131072
    ▸ Settings - Max frame size : 16384
  ▾ Stream: WINDOW_UPDATE, Stream ID: 0, Length 4
      Length: 4
      Type: WINDOW_UPDATE (8)
    ▸ Flags: 0x00
      0... .... .... .... .... .... .... .... = Reserved: 0x0
      .000 0000 0000 0000 0000 0000 0000 0000 = Stream Identifier: 0
      0... .... .... .... .... .... .... .... = Reserved: 0x0
      .000 0000 1011 1111 0000 0000 0000 0001 = Window Size Increment: 12517377
  ▾ Stream: PRIORITY, Stream ID: 3, Length 5
      Length: 5
      Type: PRIORITY (2)
    ▸ Flags: 0x00
      0... .... .... .... .... .... .... .... = Reserved: 0x0
      .000 0000 0000 0000 0000 0000 0000 0011 = Stream Identifier: 3
      0... .... .... .... .... .... .... .... = Exclusive: False
      .000 0000 0000 0000 0000 0000 0000 0000 = Stream Dependency: 0
      Weight: 200
      [Weight real: 201]
  ▸ Stream: PRIORITY, Stream ID: 5, Length 5
  ▸ Stream: PRIORITY, Stream ID: 7, Length 5
  ▸ Stream: PRIORITY, Stream ID: 9, Length 5
  ▸ Stream: PRIORITY, Stream ID: 11, Length 5
  ▸ Stream: PRIORITY, Stream ID: 13, Length 5
```

Frame (263 bytes) | Decrypted SSL (148 bytes)

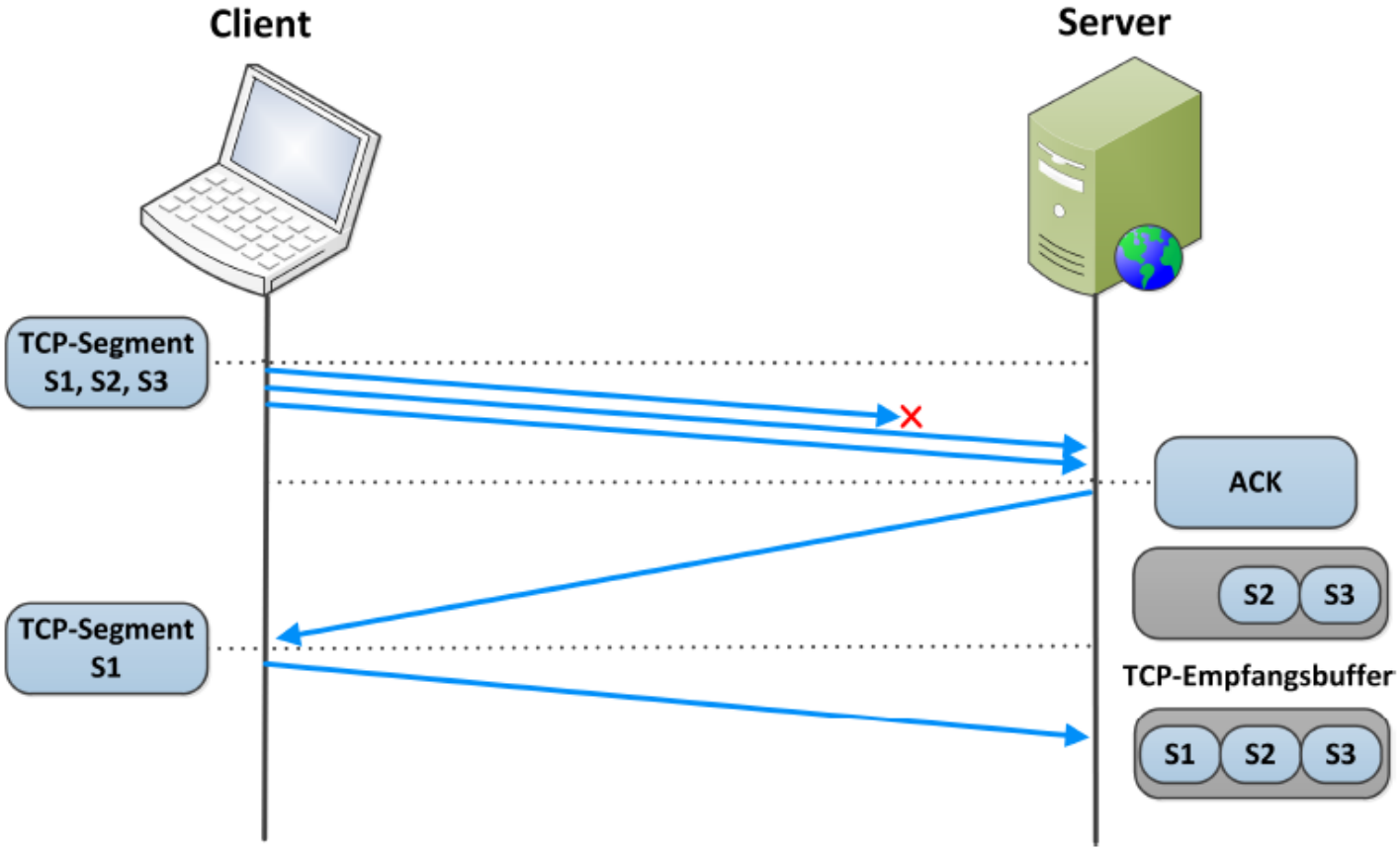
Real Weight value (Add one to value) (http2.headers.weight_real), 1 Byte

Pakete: 246 · Angezeigt: 182 (74.0%)· Verworfen: 0 (0.0%) Profil:Default

[ATV] netlab@netlab: ~ *eth0 DE 13:31

HTTP 2.0

HoL-Blocking auf TCP-Schicht



Ilya Grigorik: High Performance Browser Networking, O´ Reilly, 2014

© Martina Kannen

Literatur

Badach, Anatol, Rieger, Sebastian, Schmauch Matthias: Web-Technologien – Architekturen, Konzepte, Trends. Hanser Verlag 2003.

Becker, Markus: Stream Control Transmission Protocol -Einsatz in der Praxis anhand eines Webservers, Abschlussarbeit BCS Hochschule Bonn-Rhein-Sieg, Fachbereich Informatik, September 2009.

Belshe M. (von BitGo), Peon R. (v. Google), M. Thomson, Ed. (v. Mozilla): Hypertext Transfer Protocol Version 2 (HTTP/2), RFC 7540, Mai 2015.

Berners-Lee, Tim: The World Wide Web: A very short personal history. http://www.w3.org/People/Berners-Lee/ShortHistory.html

Bishop, M.: Hypertext Transfer Protocol (HTTP) over QUIC. draftietf-quic-http-02 (Internet-Draft). https://tools.ietf.org/html/draft-ietf-quic-http-02. Version: März 2017 (Internet-Draft)

Brück, Andreas: Vergleich von QUIC (Quick UDP Internet Connections) mit HTTP1.1 und HTTP2.0 – Messung und Analyse der Performance, Master Thesis, Mai 2017.

Comer, Douglas E: TCP/ IP –Konzepte, Protokolle und Architekturen. mitp-Verlag 2003.

Literatur

Dippel Oliver: Quantitative Untersuchungen im Internet - Optimierungsmethoden für Internet-Protokolle. Dissertation, Fachbereich Mathematik und Informatik der Philipps-Universität, Marburg, 2000.

Fielding R., Gettys, j., Mogul J.C., Masinter L., Leach P. Frystyk, H., Berners-Lee, T.: Hypertext Transfer Protocol – HTTP 1.1, RFC 2616, 1999.

Google Developers: Web metrics: Size and number of resources, https://developers.google.com/speed/articles/web-metrics, 28.3.2012

Ilya Grigorik: High Performance Browser Networking, O´ Reilly, 2014

Ilya Grigorik, Google: SPDY, HTML 5 Developer Conference, San Francisco, 15.-16. Oktober 2012, http://www.youtube.com/watch?v=SWQdSEferz8

Hachtkemper, Manuel: Analyse ausgewählter Aspekte von HTTP/2 Client-Implemetierungen. Master Thesis, 2018.

Heidemann, John, Obraczka, Katia, Touch, Joe: Modeling the Performance of HTTP Over Several Transport Protocols. IEEE/ ACM Transactions on Networking, 5(5), 1997.

Kang, Sukwoo, Fields Matthew: Experimental Study of the SCTP compared to TCP, Computer Communications and Networking Fall, Project Report, Electrical Engineering Department, 2003.

© Martina Kannen

Literatur

Leischner, Martin: Vorlesung Internetkommunikation WS 10/11.

Nielsen, Hendrik Frystyk, Gettys, Jim, Baird-Smith, Anselm, Prud'hommeaux, Eric, Håkon Wium Lie, Lilley, Chris: Network Performance Effects of HTTP/1.1, CSS1, and PNG, W3C, NOTE 24-June 1997.

QUIC Tutorial - A New Internet Transport. Internet Engineering Task Force, https://www.ietf.org/proceedings/98/slides/slides-98-edu-sessf-quic-tutorial-00.pdf. Version: März 2017.

Spero, Simon E: Analysis of HTTP Performance Problems. http://www.w3.org/Protocols/HTTP/1.0/HTTPPerformance.html, Juli 1994.

Tanenbaum, Andrew: Computernetzwerke. Pearson Studium, 2003.

Tim Berners-Lee: Long Live the Web: A Call for Continued Open Standards and Neutrality, http://www.scientificamerican.com/article.cfm?id=long-live-the-web, 22. November 2010.

Wilde, Eric:World Wide Web, Technische Grundlagen: Springer 1999.

Wintermeyer, Stefan: Schneller Wechsel, IX 8, August 2015

Literatur

Wu, Peter: SSL/TLS Decryption, SharkFest ´17 Europe,
https://sharkfesteurope.wireshark.org/assets/presentations17eu/15.pdf, 8. November 2017

Philipp Zeder: Domain-Sharding: Eine Technik die Sie 2016 vergessen können,
https://www.cyon.ch/blog/Domain-Sharding , 8. Januar 2016