Jhannes Ernesto Reimann

Task 1: Reading and Reasoning

Why is it difficult to develop and deploy/adopt new transport protocols?

It's hard to get any new transport protocol past today's networks because they're laced with "middleboxes" (NATs, firewalls, HTTP proxies, traffic scrubbing devices, performance-enhancers) that routinely strip or rewrite anything they don't recognize. These boxes will silently remove unknown TCP options (so your new handshake never makes it into the data phase), rewrite initial sequence numbers (so you can't safely embed absolute byte-counters), enforce fully contiguous byte streams (so any deliberate "holes" for partial reliability or multipath will be filled or blocked), and even split or coalesce packets (by TCP proxies or NIC offloads), breaking any tight binding between metadata and the original segment. On top of that, HTTP-aware gateways on ports 80/443 will often terminate or proxy-rewrite your connection entirely, preventing any end-to-end innovation unless it's wrapped in plain HTTP.

Because you can't change the network infrastructure, any new transport feature must be squeezed into the narrow confines of existing TCP behavior. In practice that means you have to negotiate your feature in the very first SYN exchange (and reliably detect if it's been stripped), fall back immediately to vanilla TCP whenever something goes awry, use relative sequence-offsets rather than absolute numbers, tolerate option duplication or loss due to segmentation offload, forbid any non-contiguous sequence gaps, and guard against "normalizers" that might replay or rewrite retransmissions—all while still delivering a correct, in-order byte stream if your feature isn't supported . These constraints make designing, deploying and ultimately getting people to adopt any genuinely new transport protocol a real challenge.

How does QUIC overcome the ossification issue?

QUIC encrypts and authenticates nearly its entire packet header so that on-path middleboxes cannot inspect or modify transport-layer fields, preventing them from "hard-coding" quirks of earlier versions and thus avoiding the decades-long ossification that TCP has suffered.

Why does it use UDP?

By encapsulating its own transport protocol in UDP datagrams, QUIC can be implemented entirely in user-space — avoiding slow OS kernel upgrades — and can traverse existing middleboxes without requiring them to understand a new IP-protocol number, since UDP is already widely permitted.

What features and benefits does it provide over TCP?

- fuses its cryptographic and transport handshakes to enable 0-RTT connection setup on repeat connections,
- multiplexes independent bidirectional streams over one connection to eliminate head-of-line blocking,
- uses per-packet monotonically increasing packet numbers and richer ACK ranges for more accurate loss detection,
- supports seamless connection migration via a 64-bit Connection ID when clients change IP addresses, and
- provides built-in encryption, per-stream flow control and a pluggable congestion-control interface — all in user-space for rapid iteration and deployment.

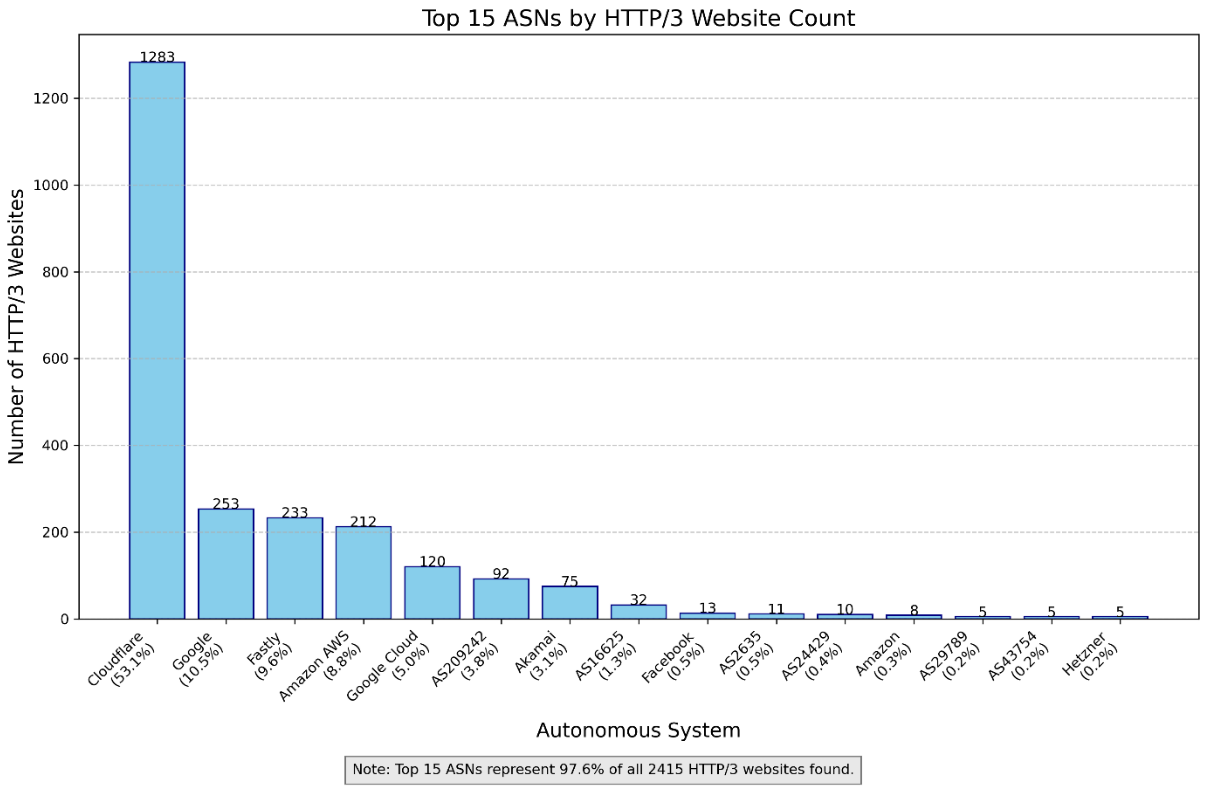
Task 2: QUIC Adoption

Make sure to convert the timings to reasonable units (e.g., ms or s), if necessary. Briefly explain what each of the above metrics represents. Save your measurement results in a CSV file (with domain name + above metrics as columns). Submit your description, CSV file, and measurement script(s).

• **time_namelookup**:	Time from the start until the domain name resolution is completed (Domain → IP)
• **time_connect**:	Time from the start until the TCP connection to the server is established (till TCP-Handshake)
• **time_appconnect**:	Time from the start until the SSL/TLS handshake is completed (includes connection time)
• **time_pretransfer**:	Time from the start until curl is ready to begin the transfer
• **time_starttransfer**:	Time from the start until the first byte of the response is received (TTFB)
• **time_redirect**:	Time spent on redirection steps
• **time_total**:	Total Time from start to finish of the request
• **remote_ip**:	IP address of the server
• **remote_port**:	Port number on the server

Jhannes Ernesto Reimann

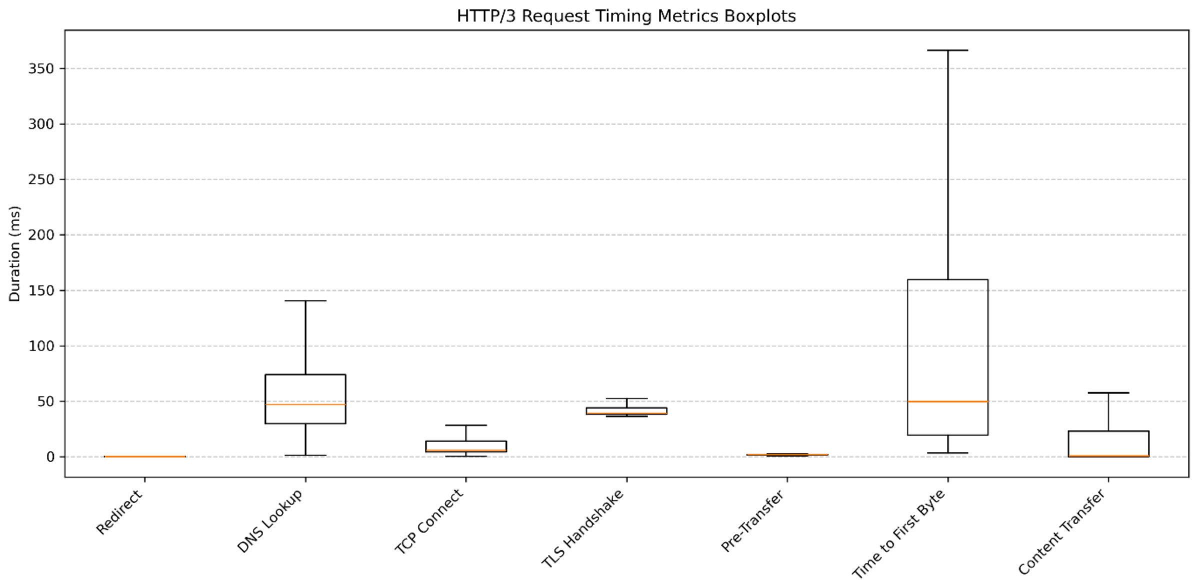
How many of the 10k websites support HTTP/3? Determine the AS numbers for the IP addresses that you collected. Count the number of websites for each ASN, sort them in descending order, and export it to a CSV file (columns: asn,num_http3_websites). Draw a bar chart using this CSV file. Which ASNs have the highest number of HTTP/3 enabled websites?, i.e., who is responsible for pushing HTTP/3 adoption? Submit your answers, scripts, CSV file and plot as PDF.



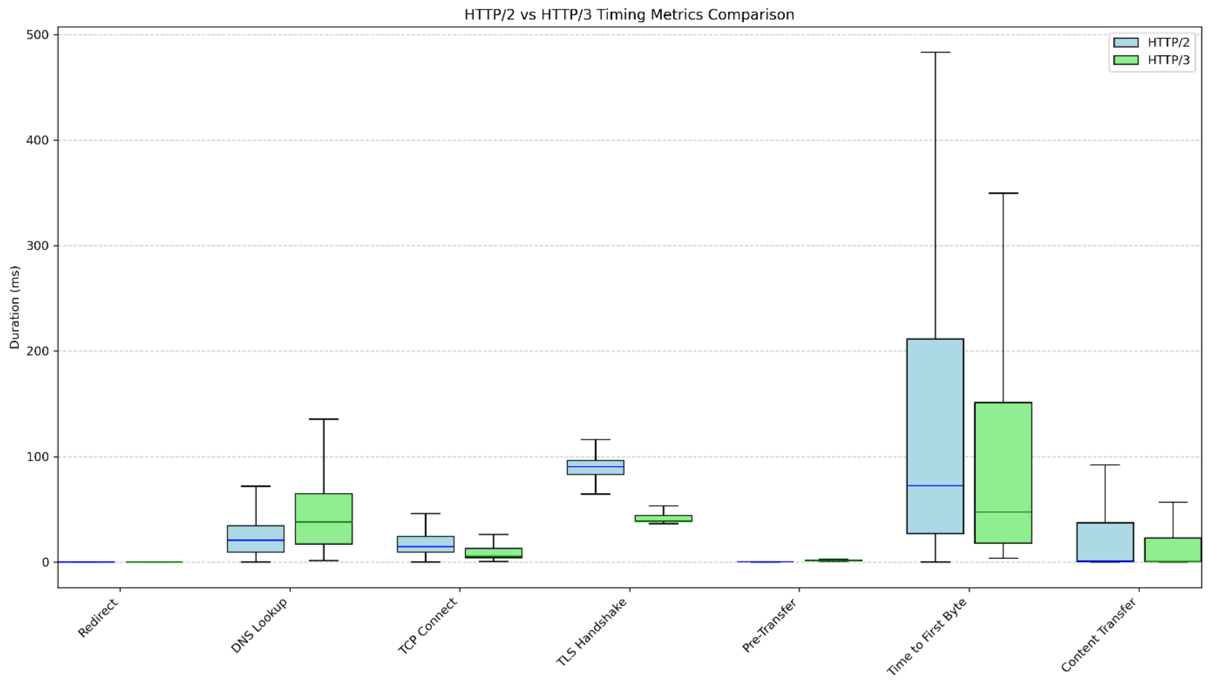
- 2415 websites support HTTP/3 out of the 10k websites. (depending on date of download of the tranco list)
- Cloudflare has the highest number of HTTP/3 enabled websites. The entities most responsible for pushing HTTP/3 adoption in practice are Cloudflare, Google, and Fastly, as they serve a significant portion of the global web traffic. Their early and widespread support ensures that even small website owners can offer HTTP/3 with minimal configuration, driving mass adoption.

Task 3: HTTP/3 Latency

Order the collected timing metrics by the sequential order of their corresponding events (i.e., DNS lookup happens before y, etc.). Calculate the starting/end times and duration of the events; note that some metrics include other metrics, so you will need to calculate deltas in some cases. Draw boxplots over all measurements for each of the events/metrics (cf. Assignment 02); you can use the description of the events/metrics as labels for your x-axis (e.g., DNS lookup, ...). Submit your answers/descriptions, analysis and plotting scripts, and plot.



In order to compare the performance of QUIC+HTTP/3, repeat your curl measurements for the Top 1k websites which you have found to support HTTP/3 with TCP+TLS+HTTP/2. If you found less than 1k websites with HTTP/3 support, use this number instead. Extend your scripts from the previous task so that you show the distributions for HTTP/2 and HTTP/3 next to each other. Which distributions are similar (i.e., unaffected) and which boxes are different (i.e., faster/slower) when comparing HTTP/2 and HTTP/3? Submit your extended script(s), plot, and answers.



Similar Distributions (Relatively Unaffected):

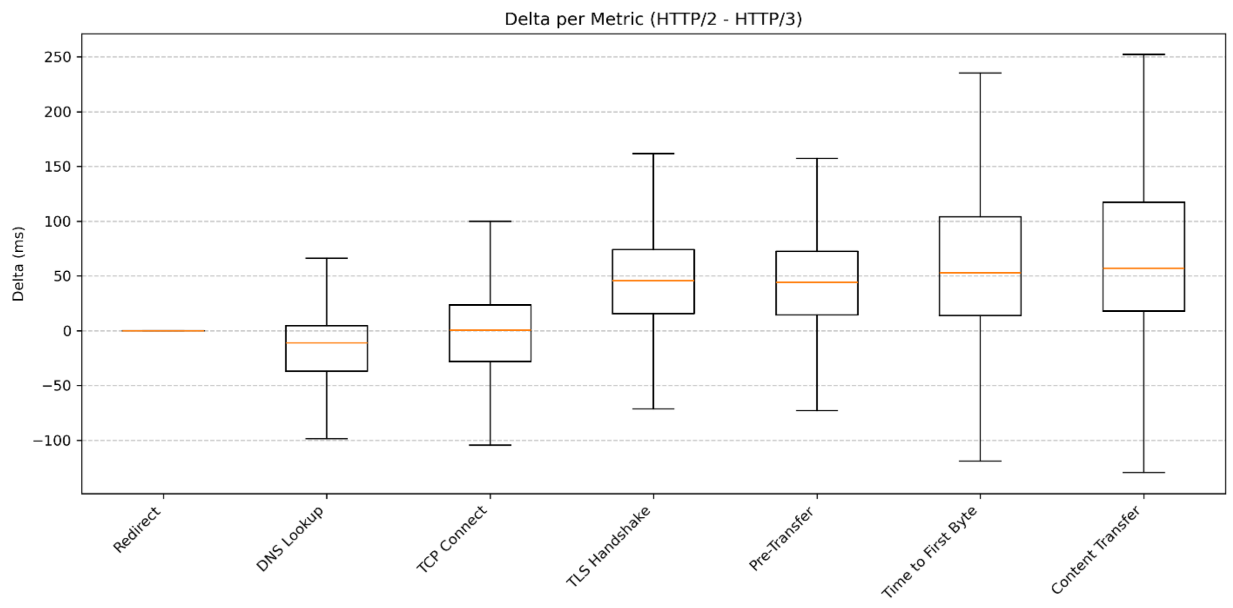
- **Redirect**: Both HTTP/2 and HTTP/3 show minimal time spent on redirects, with very similar distributions near zero.
- **Pre-transfer**: Both protocols show very small values with similar distributions.

Different Distributions (Faster/Slower):

- **DNS Lookup**: HTTP/3 (green) shows higher median and wider distribution compared to HTTP/2 (blue), indicating HTTP/3 is generally slower for DNS lookups.
- **TCP Connect**: HTTP/2 has slightly higher median values than HTTP/3. This is expected since HTTP/3 uses QUIC which doesn't have a traditional TCP connection phase.
- **TLS Handshake**: HTTP/2 shows significantly higher values (median around 90ms) compared to HTTP/3 (median around 40ms). This is one of the most dramatic differences, with HTTP/3 being much faster in TLS handshaking.
- **Time to First Byte**: Both protocols show wide distributions, but HTTP/2 has a higher median and more outliers extending to nearly 500ms, while HTTP/3 generally shows lower values.
- **Content Transfer**: HTTP/2 shows slightly higher median values than HTTP/3, suggesting HTTP/3 may be marginally faster for content transfer.

Jhannes Ernesto Reimann

Calculate the differences in raw timing metrics from curl between TCP+TLS+HTTP/2 and QUIC+HTTP/3 for each of the (max.) 1k websites, i.e., Δvalue = value_http2 − value_http3. For each metric, draw a boxplot over the deltas. Store the 25th, 50th, and 75th percentiles of each box in a CSV file. What are the interquartile range of the deltas, and what does that mean w.r.t. HTTP/2 and HTTP/3? Submit your analysis/plotting script(s) and discussion.



1. **Redirect**: IQR = 0.0 - 0.0 = 0.0 ms
   - Meaning: No difference between HTTP/2 and HTTP/3 for redirects.
2. **DNS Lookup**: IQR = 4.61 - (-36.9325) = 41.5425 ms
   - Meaning: High variability in DNS lookup performance differences, with HTTP/3 generally slower (negative median delta).
3. **TCP Connect**: IQR = 23.68 - (-27.965) = 51.645 ms
   - Meaning: Very wide variability in connection establishment, with a median slightly favoring HTTP/2.
4. **TLS Handshake**: IQR = 74.155 - 15.7375 = 58.4175 ms
   - Meaning: Significant variability, but consistently faster in HTTP/3 (positive deltas).
5. **Pretransfer**: IQR = 72.5125 - 14.3675 = 58.145 ms
   - Meaning: Similar to TLS handshake, consistently faster in HTTP/3 with high variability.
6. **Time to First Byte**: IQR = 103.9775 - 13.92 = 90.0575 ms
   - Meaning: Very large variability, but generally faster in HTTP/3.
7. **Content Transfer** (time_total): IQR = 117.2675 - 18.0975 = 99.17 ms
   - Meaning: Extremely wide variability, with HTTP/3 generally performing better.

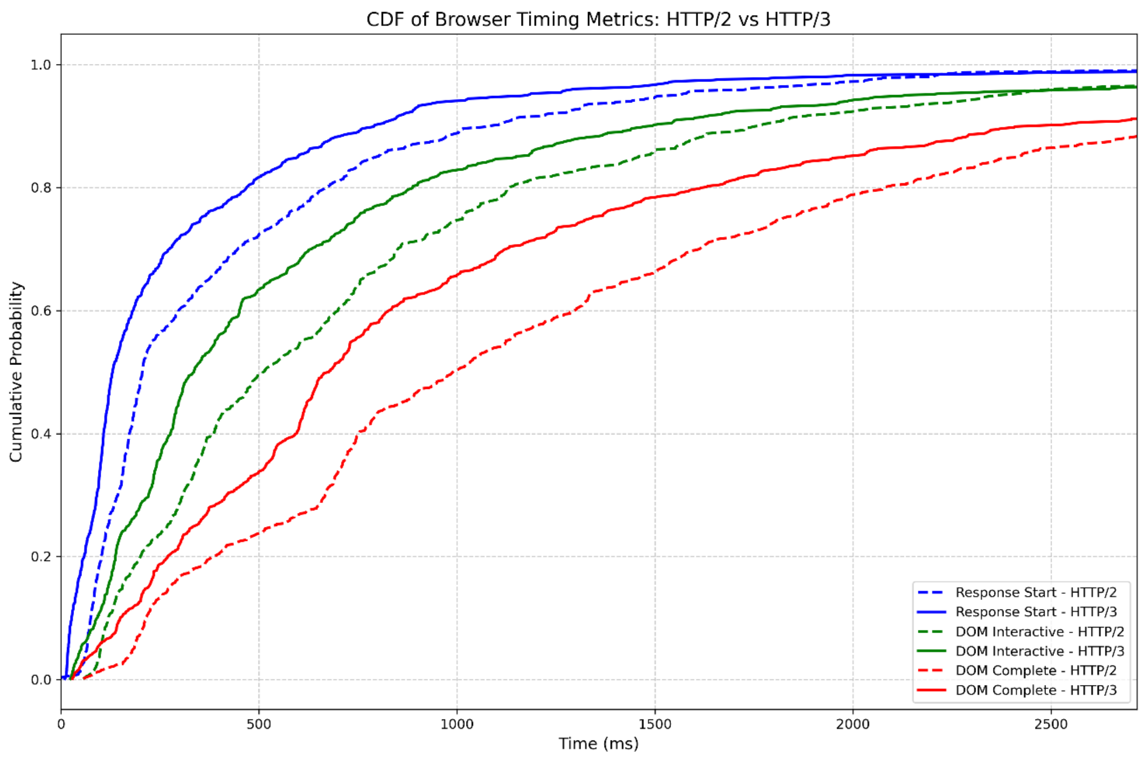
What this means regarding HTTP/2 and HTTP/3:

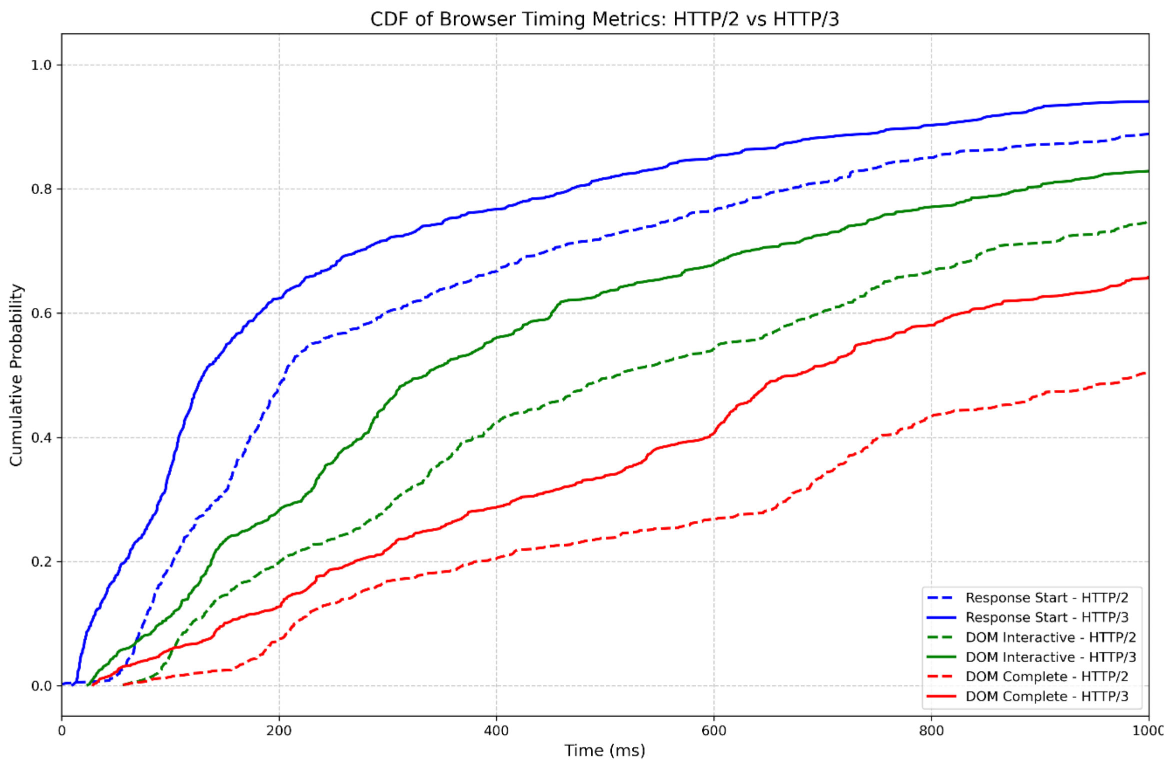
- The large IQRs across most metrics indicate high variability in performance differences between the protocols. This suggests that performance advantages are not consistent across all websites.
- DNS lookup shows negative median delta, meaning HTTP/3 is typically slower for DNS resolution.
- TLS handshake, pretransfer, time to first byte, and total transfer time all show positive median deltas, indicating HTTP/3 is generally faster in these aspects.

- The widest IQRs are in the time to first byte and content transfer metrics, suggesting that the greatest performance variability occurs during data transfer phases.
- This high variability aligns with previous findings that HTTP/3 performance can be affected by factors like first-connection overhead in QUIC, network optimization favoring TCP over UDP, less mature HTTP/3 server implementations, and potential middlebox interference with UDP traffic.

Task 4: HTTP/3 Rendering Time

Draw a CDF figure for both HTTP/2 (dashed lines) and HTTP/3 (solid lines) measurements, with separate curves for responseStart, domInteractive, and domComplete. What do the different metrics represent? For some percentiles of your choice, discuss the difference between HTTP/2 and HTTP/3, esp. regarding the impact of HTTP/3 and QUIC for the user. Submit your plotting script, plot, and discussion.





Metrics Explained

- **Response Start**: Time until the browser receives the first byte of the response. This measures network latency and server processing time.
- **DOM Interactive**: Time until the HTML document has been fully parsed and the browser can begin interacting with the page. This is when scripts can start executing.
- **DOM Complete**: Time until all processing of the HTML document and subresources (images, stylesheets) is finished. This is when the page is fully loaded.

I have measured this 2 times in total. 1 time with a chrome option "--quic-version" and trying different HTTP/3 QUIC versions until i got a valid response. The other time (after I saw your Moodle Forum post) with the option "--origin-to-force-quic-on=www.example.org:portnumber". I got different amounts of verified protocols and very different deviations. I will list both results and both scripts numbered with _quic-version and _origin-to-force-quic-on.

--quic-version

Summary Statistics (in milliseconds):

Metric	Protocol	Count	Median	Mean	P25	P75
responseStart	HTTP/2	704	205.5	442.1	119.3	565.1
responseStart	HTTP/3	704	130.7	359.3	77.0	353.4
domInteractive	HTTP/2	704	512.5	795.9	269.1	1014.2
domInteractive	HTTP/3	704	335.9	787.6	171.4	743.8
domComplete	HTTP/2	704	990.3	1363.7	540.8	1823.4
domComplete	HTTP/3	704	676.8	1399.0	341.4	1339.1

25th Percentile (Faster Connections)

- **responseStart**: HTTP/3 is 42.3ms faster (119.3ms vs 77.0ms)
- **domInteractive**: HTTP/3 is 97.7ms faster (269.1ms vs 171.4ms)
- **domComplete**: HTTP/3 is 199.4ms faster (540.8ms vs 341.4ms)

For faster connections, HTTP/3 shows significant advantages across all metrics, with the most substantial improvement in complete page load time.

50th Percentile (Median)

- **responseStart**: HTTP/3 is 74.8ms faster (205.5ms vs 130.7ms)
- **domInteractive**: HTTP/3 is 176.6ms faster (512.5ms vs 335.9ms)
- **domComplete**: HTTP/3 is 313.5ms faster (990.3ms vs 676.8ms)

At the median, HTTP/3's advantage is even more pronounced, with over 300ms faster page load times.

75th Percentile (Slower Connections)

- **responseStart**: HTTP/3 is 211.7ms faster (565.1ms vs 353.4ms)
- **domInteractive**: HTTP/3 is 270.4ms faster (1014.2ms vs 743.8ms)
- **domComplete**: HTTP/3 is 484.3ms faster (1823.4ms vs 1339.1ms)

For slower connections, HTTP/3 demonstrates its greatest advantage, with nearly half a second faster complete page loads.

Mean Values

- **responseStart**: HTTP/3 is 82.8ms faster (442.1ms vs 359.3ms)
- **domInteractive**: HTTP/3 is 8.3ms faster (795.9ms vs 787.6ms)
- **domComplete**: HTTP/3 is 35.3ms slower (1363.7ms vs 1399.0ms)

Interestingly, while HTTP/3 maintains its advantage in initial response time, the mean values for complete page loads show HTTP/3 slightly slower. This suggests some outlier cases where HTTP/3 performance degrades significantly.

Impact of HTTP/3 and QUIC for Users

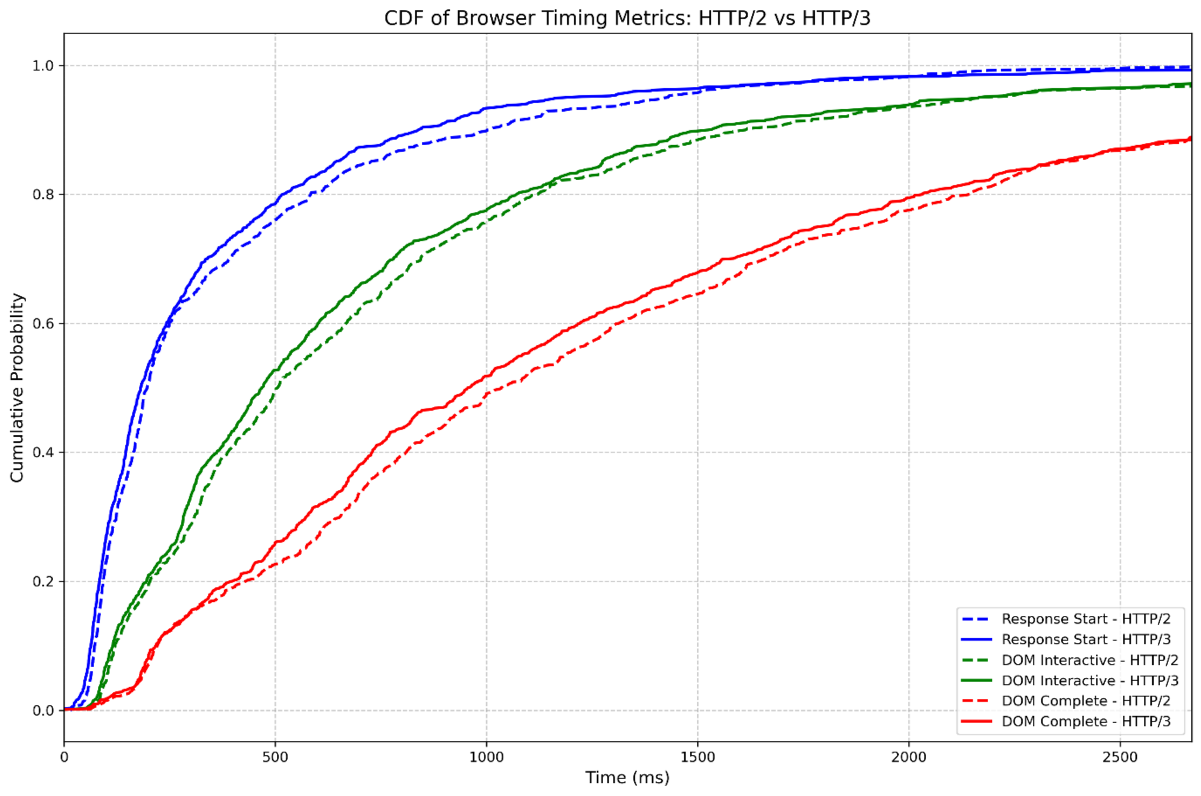
Positive Impacts

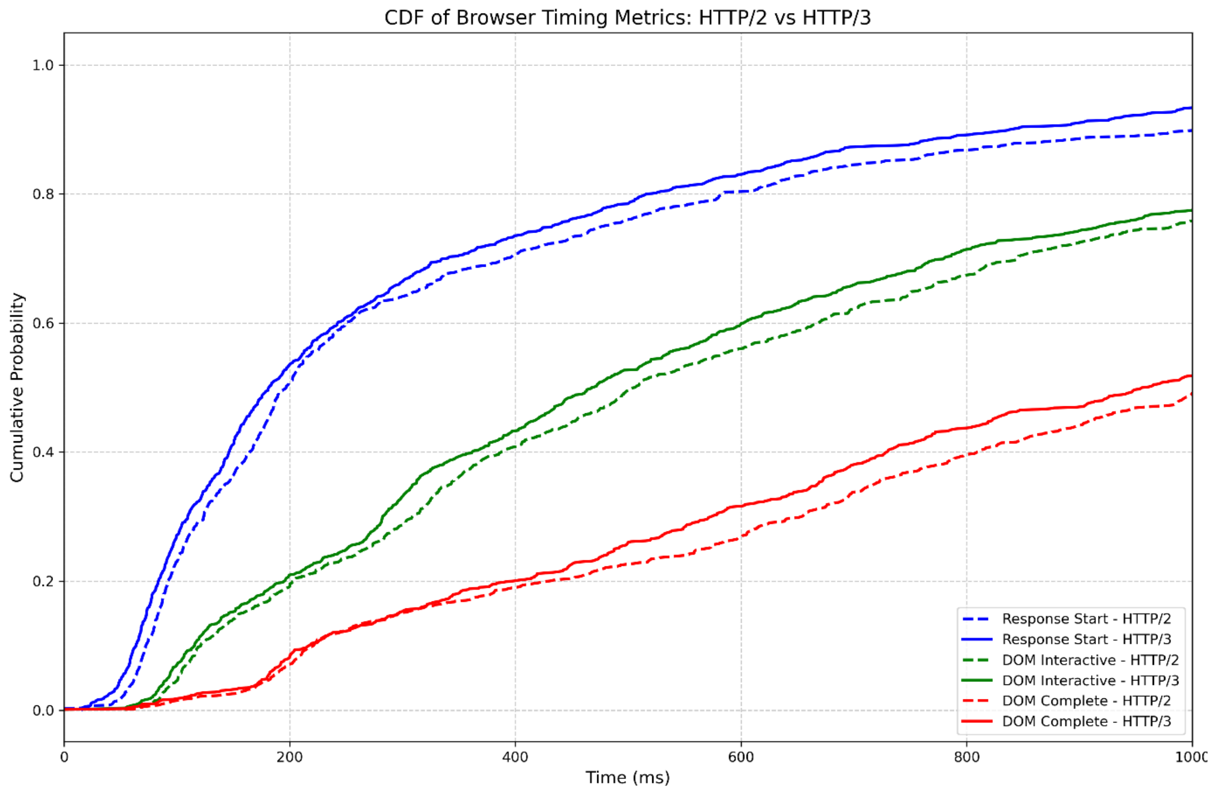
- **Faster Initial Response**: The consistently lower responseStart times (42-212ms faster across percentiles) mean users see content appearing more quickly with HTTP/3.
- **Improved Interactivity**: The significant reduction in domInteractive times (98-270ms faster) means users can start interacting with pages sooner.
- **Better Performance on Slower Connections**: The most substantial improvements are at the 75th percentile, suggesting HTTP/3's connection migration and loss recovery features benefit users with less reliable connections.
- **Perceptible Improvements**: Research shows that improvements of 100ms or more are perceptible to users. The median improvement of 313.5ms for complete page loads exceeds this threshold significantly.

Considerations

- **Outlier Performance**: While HTTP/3 is faster at all percentiles, the mean domComplete time is slightly worse, indicating some scenarios where HTTP/3 significantly underperforms.
- **Protocol Maturity**: These results contradict some earlier findings about HTTP/3 performance, suggesting that browser and server implementations have matured significantly.
- **User Experience Impact**: The improvements in responseStart and domInteractive are particularly important for perceived performance - users will feel pages are loading faster even before they're completely loaded.

--origin-to-force-quic-on





Summary Statistics (in milliseconds):

Metric	Protocol	Count	Median	Mean	P25	P75
responseStart	HTTP/2	888	196.8	389.8	105.5	478.3
responseStart	HTTP/3	888	181.7	357.7	95.6	432.9
domInteractive	HTTP/2	888	506.2	752.7	268.3	983.7
domInteractive	HTTP/3	888	468.9	717.3	251.0	919.7
domComplete	HTTP/2	888	1035.9	1407.7	570.3	1888.0
domComplete	HTTP/3	888	958.6	1340.6	490.9	1792.6

25th Percentile (Faster Connections)

- **responseStart**: HTTP/3 is 10.0ms faster (105.5ms vs 95.6ms)
- **domInteractive**: HTTP/3 is 17.3ms faster (268.3ms vs 251.0ms)
- **domComplete**: HTTP/3 is 79.4ms faster (570.3ms vs 490.9ms)

For faster connections, HTTP/3 shows modest advantages across all metrics, with the most substantial improvement in complete page load time.

50th Percentile (Median)

- **responseStart**: HTTP/3 is 15.1ms faster (196.8ms vs 181.7ms)
- **domInteractive**: HTTP/3 is 37.3ms faster (506.2ms vs 468.9ms)
- **domComplete**: HTTP/3 is 77.3ms faster (1035.9ms vs 958.6ms)

At the median, HTTP/3's advantage becomes more noticeable, with a significant improvement in page load times.

75th Percentile (Slower Connections)

- **responseStart**: HTTP/3 is 45.4ms faster (478.3ms vs 432.9ms)
- **domInteractive**: HTTP/3 is 64.0ms faster (983.7ms vs 919.7ms)
- **domComplete**: HTTP/3 is 95.3ms faster (1888.0ms vs 1792.6ms)

For slower connections, HTTP/3 demonstrates its greatest advantage, with nearly 100ms faster complete page loads.

Mean Values

- **responseStart**: HTTP/3 is 32.2ms faster (389.8ms vs 357.7ms)
- **domInteractive**: HTTP/3 is 35.4ms faster (752.7ms vs 717.3ms)
- **domComplete**: HTTP/3 is 67.1ms faster (1407.7ms vs 1340.6ms)

The mean values consistently show HTTP/3's advantage across all metrics, indicating improved overall performance without significant outliers degrading HTTP/3 performance.

Impact of HTTP/3 and QUIC for Users

Positive Impacts

- **Consistent Performance Improvement:** HTTP/3 shows consistent performance advantages across all metrics and percentiles, unlike the previous measurements where mean complete page loads were slower.
- **Faster Initial Response:** The improved responseStart times (10-45ms faster across percentiles) mean users see content appearing more quickly with HTTP/3.
- **Improved Interactivity:** The reduction in domInteractive times (17-64ms faster) means users can start interacting with pages sooner.
- **Better Performance on Slower Connections:** The most substantial improvements are at the 75th percentile, suggesting HTTP/3's connection migration and loss recovery features benefit users with less reliable connections.

Considerations

- **More Modest Gains**: While HTTP/3 is consistently faster, the performance advantages (77-95ms for complete page loads) are more modest than in the previous assessment (199-484ms).
- **Progressive Improvement**: The performance gains increase as connection quality decreases (from 25th to 75th percentile), indicating HTTP/3's resilience in challenging network conditions.
- **Protocol Maturity:** These results show HTTP/3 has matured as a protocol, offering consistent performance improvements over HTTP/2 without the performance degradation seen in previous measurements.
- **User Experience Impact:** While improvements are smaller than previously reported, they still contribute to a smoother browsing experience, especially for users on slower networks where the improvements are most significant.