

Evaluating Web and Video Content Delivery over Google QUIC

Rohit Panda

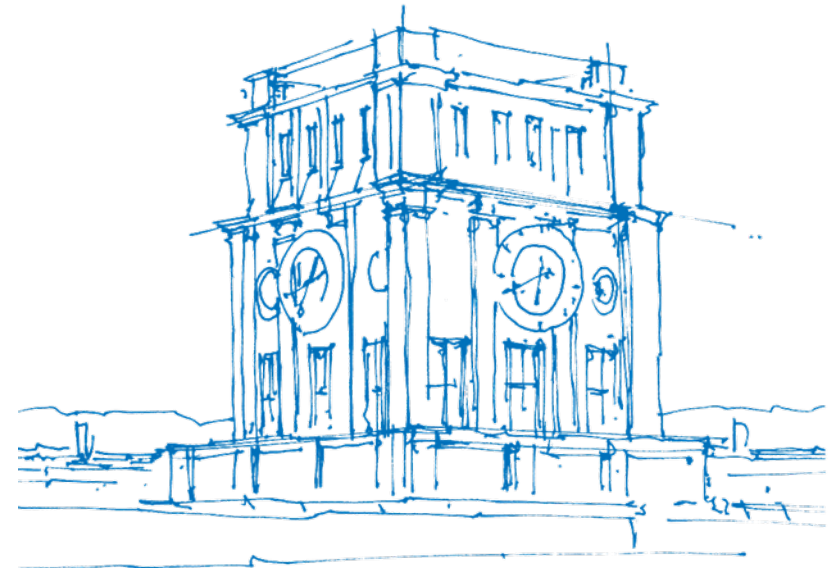
rohit.panda@tum.de

Advisor: Dr. Vaibhav Bajpai

bajpaiv@in.tum.de

Supervisor: Prof. Dr. Jörg Ott

ott@in.tum.de



Tum Uhrenturm

München, February 21, 2019

Motivation

- Web latency and security are becoming more and more important.
- Middleboxes have caused a protocol ossification. Attempted extensions to TCP like TCP Fast Open[27], SCTP[35], Multipath TCP [11] have not seen widespread deployment.
- QUIC is a user-space transport protocol built on top of UDP. Also QUIC provides for authenticated, encrypted header and payload which avoids dependency on vendors and ISPs
- Faster development and deployment cycles. Already on Q044 version and accounts for more than 35% of Google's total egress traffic and 7% of global internet traffic.
- QUIC has been shown to reduce search latency by 8% for desktop users and 3.6% for mobile users.[21]

Goal and Approach

- Initial performance results from Google showed improvements in latency, rebuffer rates and lower packet loss rates but there is still a lack of repeatable studies.
- Rapid development cycle of QUIC means that many new features are introduced in each version of QUIC and support for older versions is dropped. So previous studies(Refer ??) become obsoleted quickly.
- In this thesis we aim to evaluate the performance of QUIC(Q035, Q039, Q043, Q044) and compare it with TCP/TLS.
- To do this we conduct experiments on real scenarios rather than in an emulated environment to investigate connection establishment time, Time to first byte, download time, YouTube QoE metrics, Throughput, fairness in competing flow scenarios and CPU utilization.

Goal and Approach

- **RQ1 : How do different versions of QUIC compare with each other and TCP/TLS 1.2 and TCP/TLS 1.3 on IPv4 and IPv6?**
- **RQ2 : How do the versions of QUIC compare with each other on different Autonomous Systems?**
- **RQ3 : How does throughput and CPU utilization of QUIC compare with TCP/TLS?**
- **RQ4 : Is QUIC fair towards TCP/TLS in sharing bandwidth?**

Background

QUIC replaces most of the traditional HTTPS stack: TCP, TLS, HTTP/2.

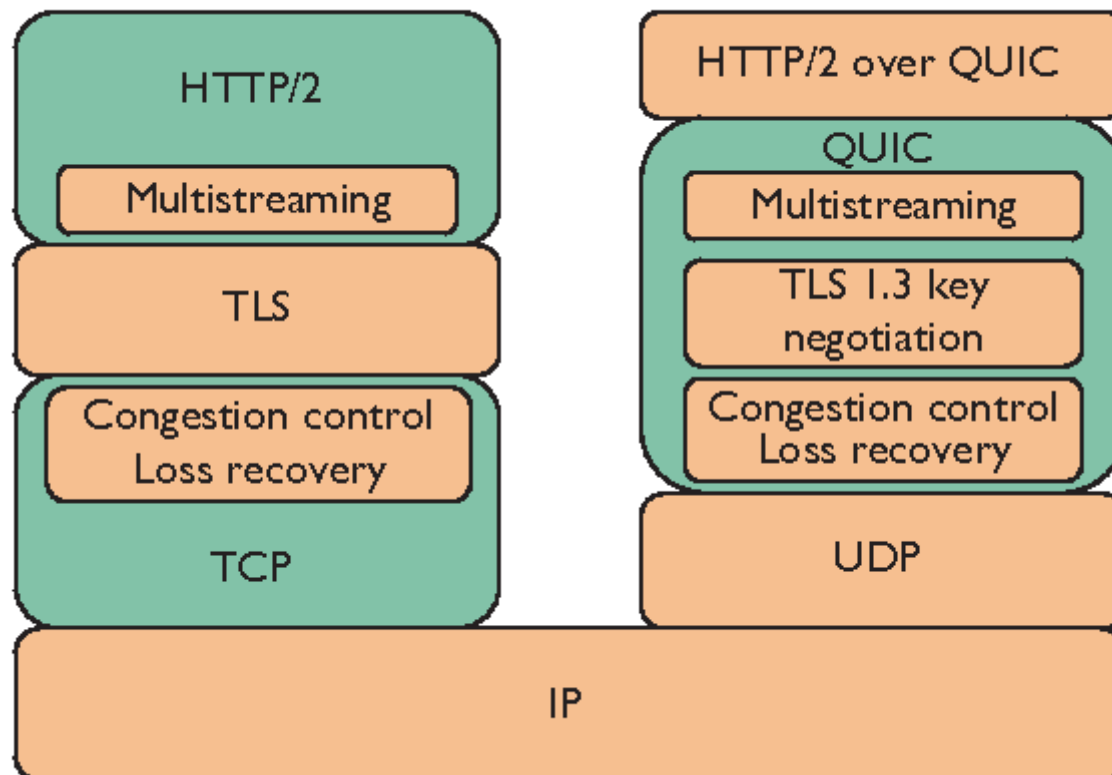


Figure: Layered QUIC architecture in the traditional HTTPS stack. QUIC incorporates features of congestion control, loss recovery similar to TCP, Multiple streams like HTTP2 and key negotiation like TLS.[7]

Methodology

- connection establishment time, time to first byte and page download time
- Analyze these metrics on AS level(announces prefix of our target websites)
- YouTube QoE metrics like Connect Time to media CDNs, Prebuffering time, Startup delay and Overall download
- Throughput and CPU utilization
- Congestion control fairness in cases of competing flows
- `perf` and `strace` to profile QUIC

Methodology

- Reused the `YouTube` test for TCP developed by S Ahsan et al.[2] which was modified to use QUIC by Sergey[24] and `tls_perf` and `quic_perf` by Bernhard[17]. Several additions and improvements were made to their tests like adding support for Q044, improving logging and error handling, adding HTTP/2 support.
- Used the latest version of `litespeed lsquic client`* and `Boringssl` library for QUIC implementation.
- `libcurl` was used for TCP implementation.
- For measurements over IETF QUIC we used the QUIC Tracker [12] test suite which supports QUIC(draft-17) and is also TLS1.3 compatible.

*<https://github.com/litespeedtech/lsquic-client/tree/1.17.8>

Measurement Setup

- Measurements were performed from 3 vantage Points.
- VM `vmott16.cm.in.tum.de` with linux OS. Bandwidth: 400 Mbits/s It supported both IPv4 and IPv6. server and 1.20 Gbits/s via TCP.
- Raspberry Pi 3 Model B with 1 GB RAM and 32 GB microSDHC Card. Bandwidth: 100 Mbits/s line. This PI was installed at a residential location
- Another Raspberry PI was installed in Bhubaneswar, India. Bandwidth: 20 Mbits/s line but peak bandwidth was measured at 14 Mbit/s during testing.
- Only IPv4 could be tested at the above two locations because of lack of IPv6 support

Connection Establishment Times

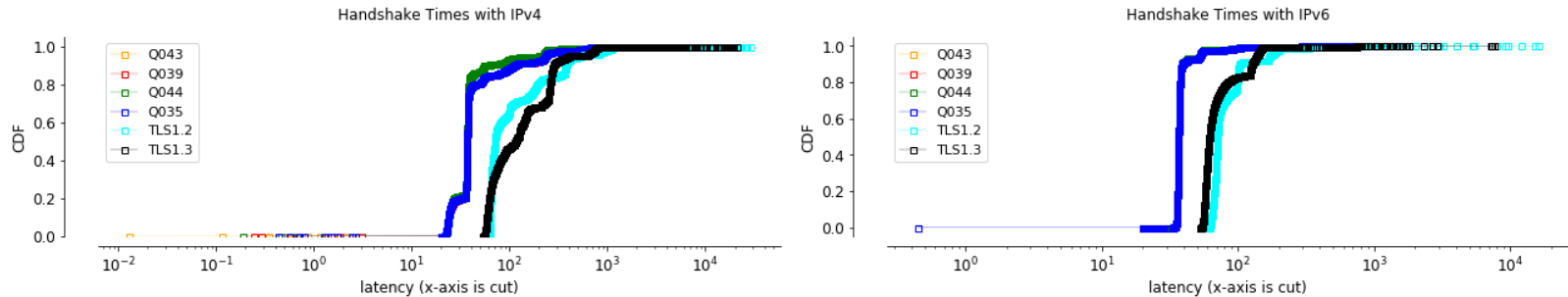


Figure: CDFs of Connection Times for QUIC(Q044, Q043, Q039, Q035) and TLS(TLS 1.3, TLS 1.2) for Ipv4, IPv6

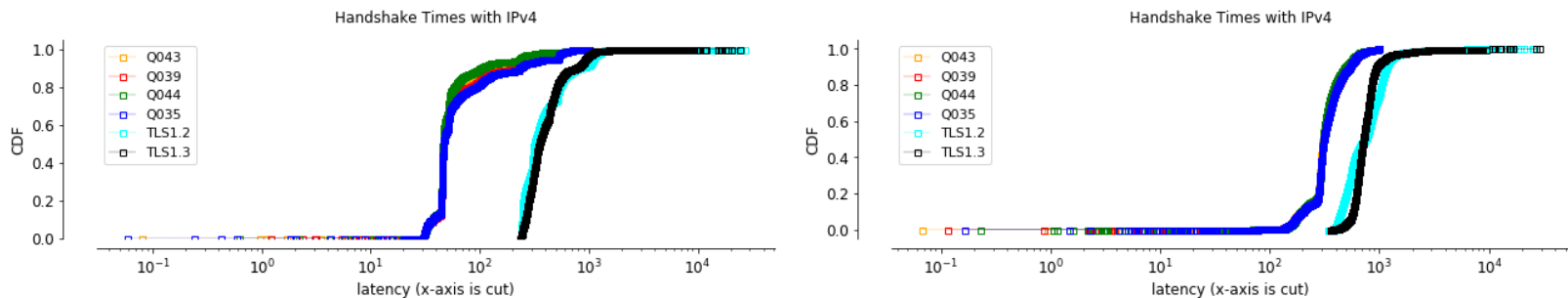


Figure: CDFs of Connection Times for QUIC(Q044, Q043, Q039, Q035) and TLS(TLS 1.3, TLS 1.2) for IPv4 from Raspberry Pi in Munich, Germany and Bhubaneswar, India

Time to first byte

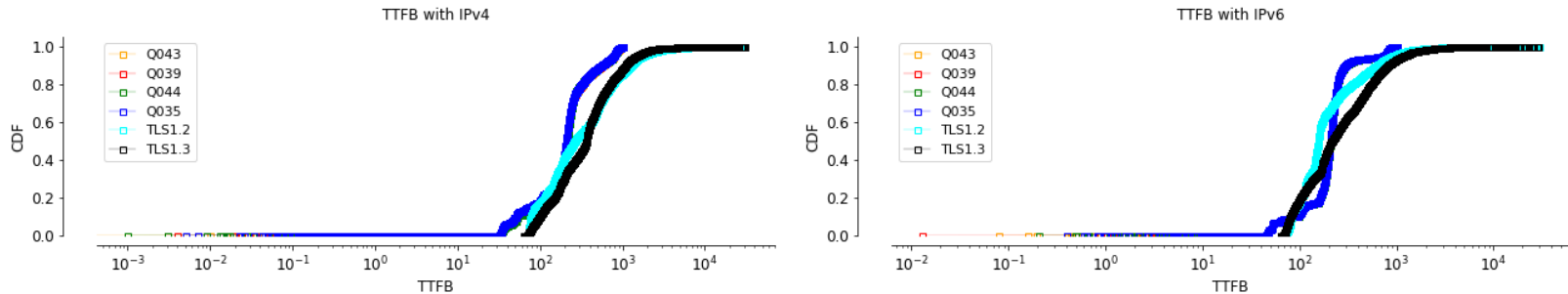


Figure: CDFs of Time to First Byte for QUIC(Q044, Q043, Q039, Q035) and TLS(TLS 1.3, TLS 1.2) for IPv4, IPv6.

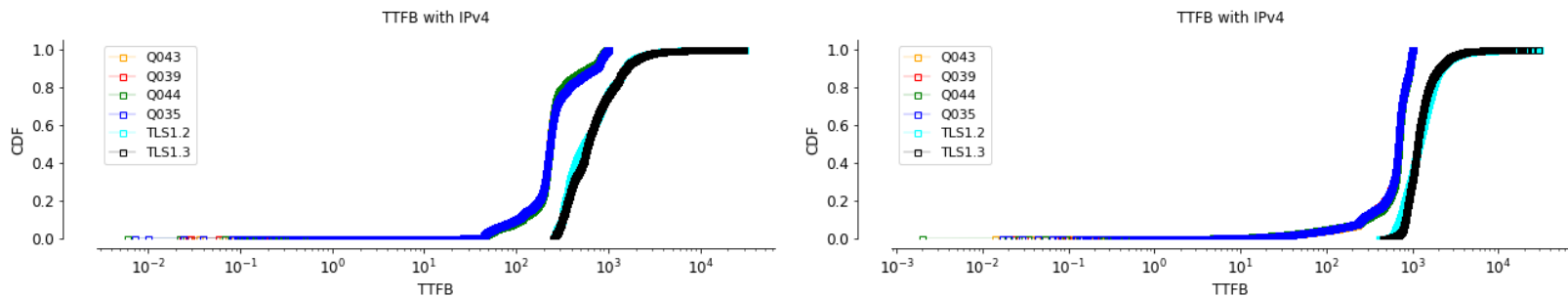


Figure: CDFs of TTFB for QUIC(Q044, Q043, Q039, Q035) and TLS(TLS 1.3, TLS 1.2) for IPv4 from Raspberry Pi in Munich, Germany and Bhubaneswar, India

AS topology

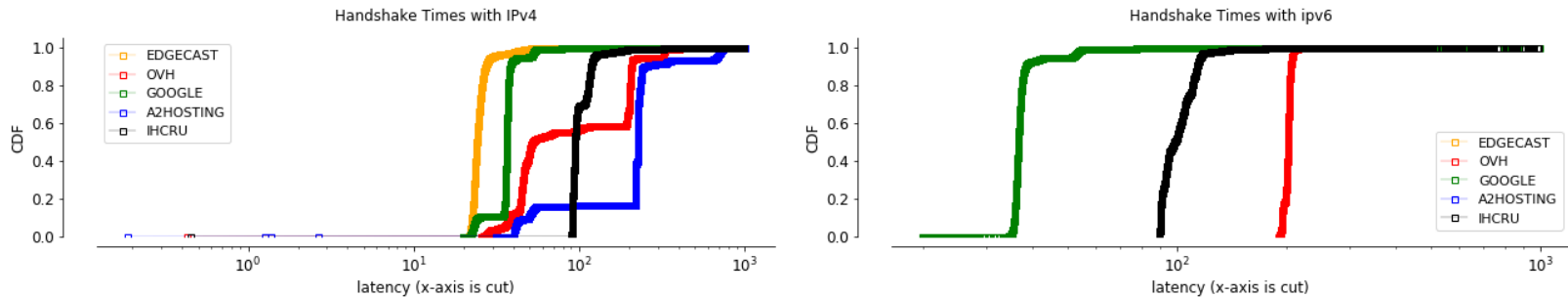


Figure: CDFs of Connection Times for all for QUIC(all versions combined) categorized by AS for IPv4, IPv6. EDGECAST performs better than GOOGLE

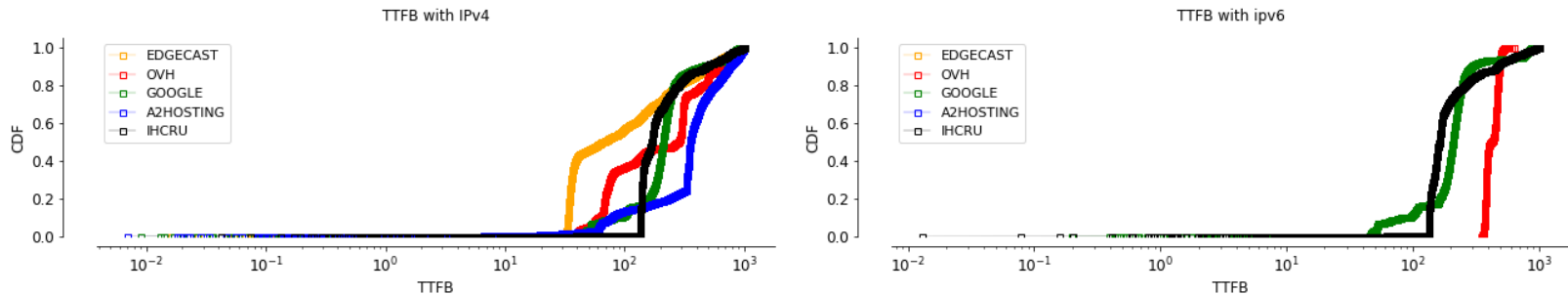


Figure: CDF of Time to First Byte for QUIC(all versions combined) categorized by AS for IPv4, IPv6

Google AS serves an overwhelming majority of websites using QUIC, almost half of the websites in our target list.

YouTube QoE

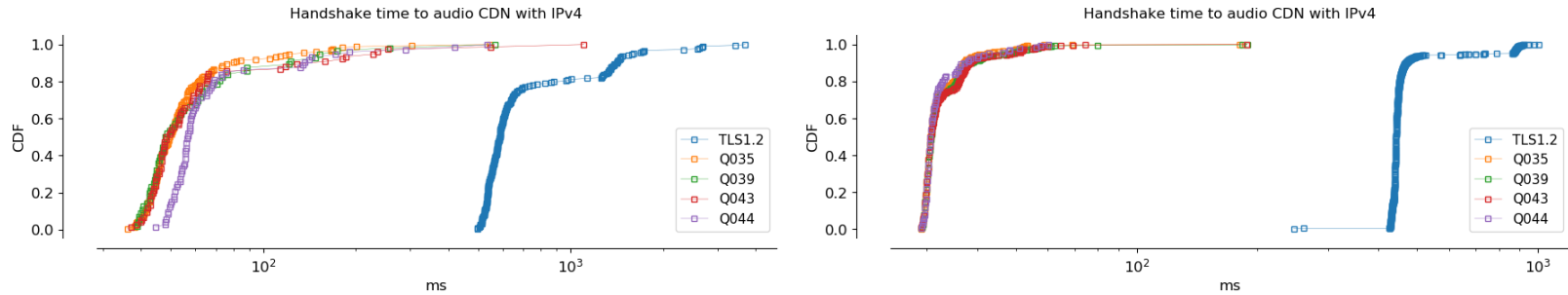


Figure: CDF of Audio Connect Time for IPv4 in Bhubaneswar, India and Munich, Germany

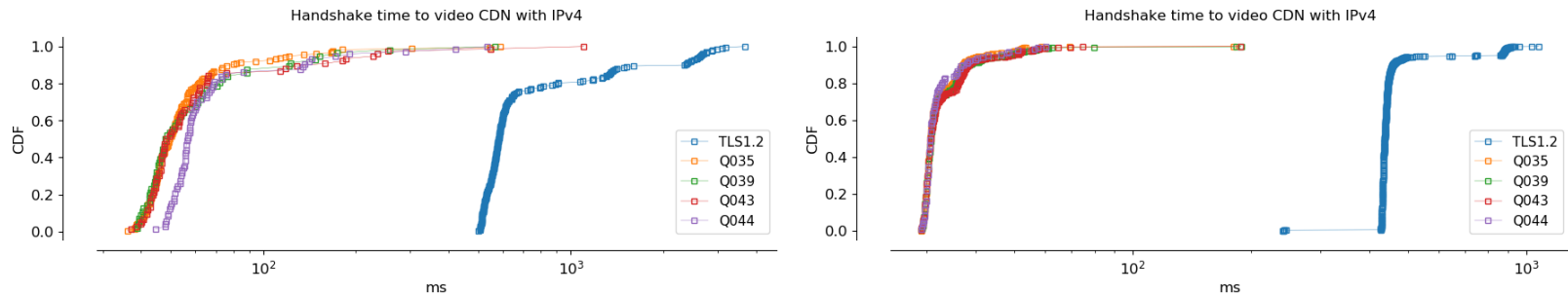


Figure: CDF of Video Connect Time for IPv4 in Bhubaneswar, India and Munich, Germany

YouTube QoE

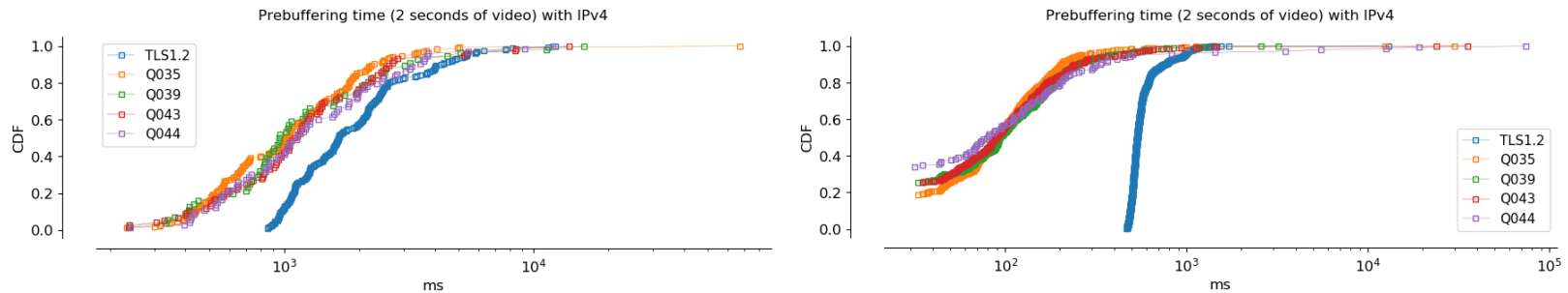


Figure: CDF of Prebuffering time for IPv4 in Bhubaneswar, India and Munich, Germany

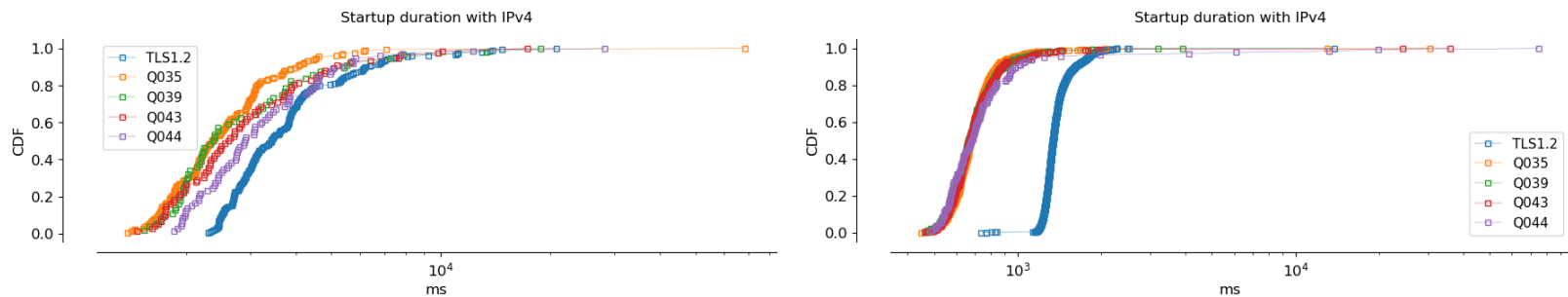


Figure: CDF of startup delay for IPv4 in Bhubaneswar, India and Munich, Germany

YouTube QoE

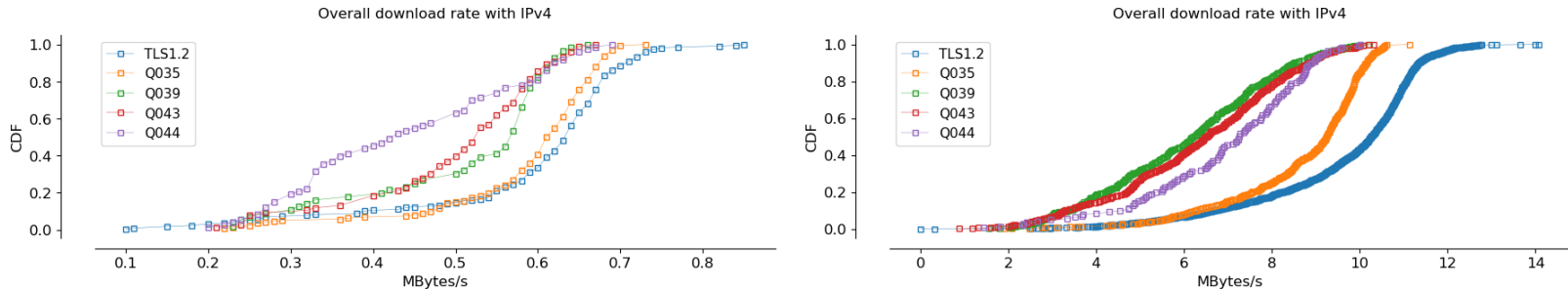


Figure: CDF of Download rate for IPv4 in Bhubaneswar, India and Munich, Germany

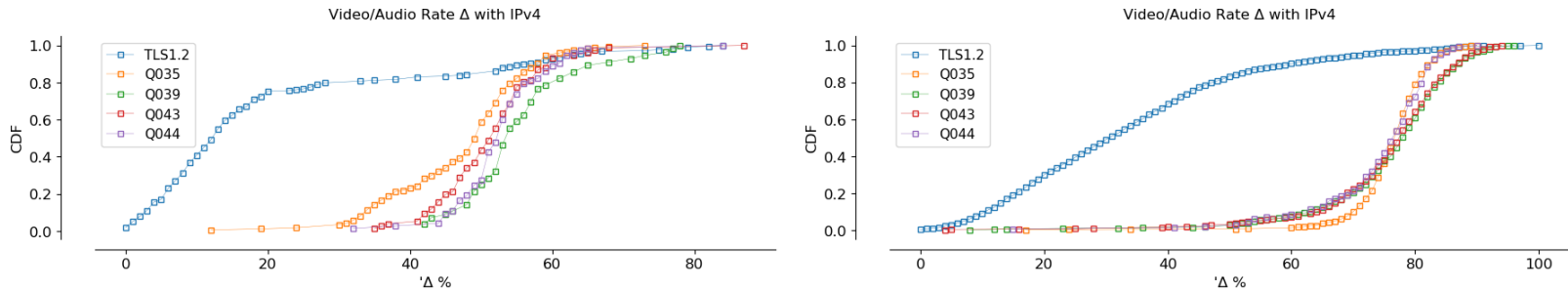


Figure: CDF of video/audio rate difference in % for IPv4 in Bhubaneswar, India and Munich, Germany

Throughput

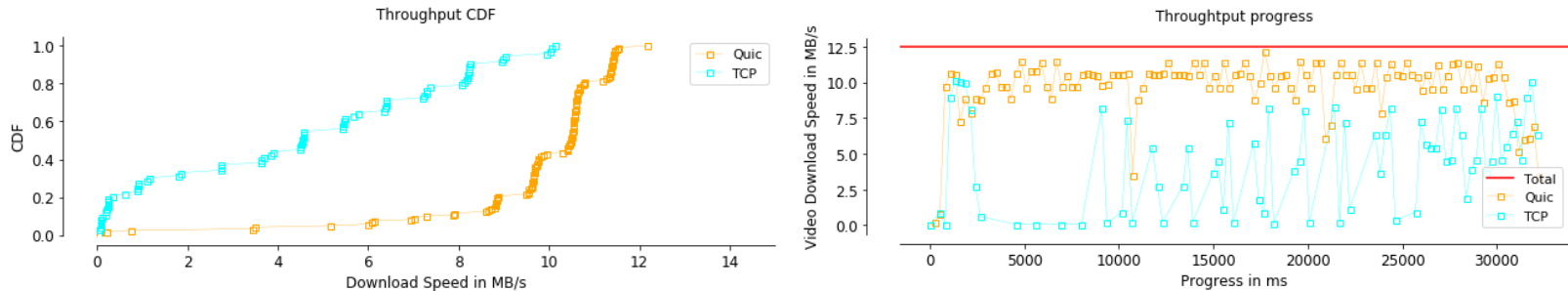


Figure: CDF of Throughput and Throughput of QUIC(Q044) and TCP downloading a YouTube video.

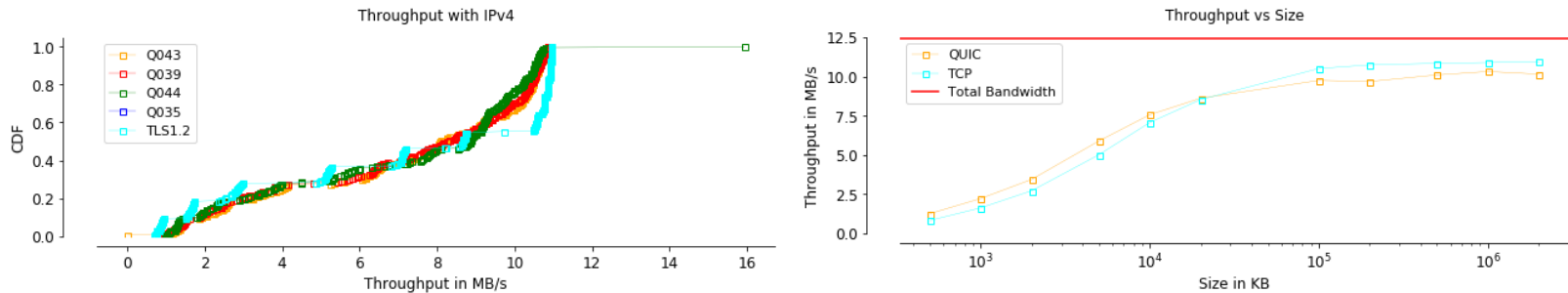


Figure: CDF of Throughput and Mean Throughput of QUIC(Q044, Q043, Q039) and TCP downloading files of different sizes from Google Drive

Fairness

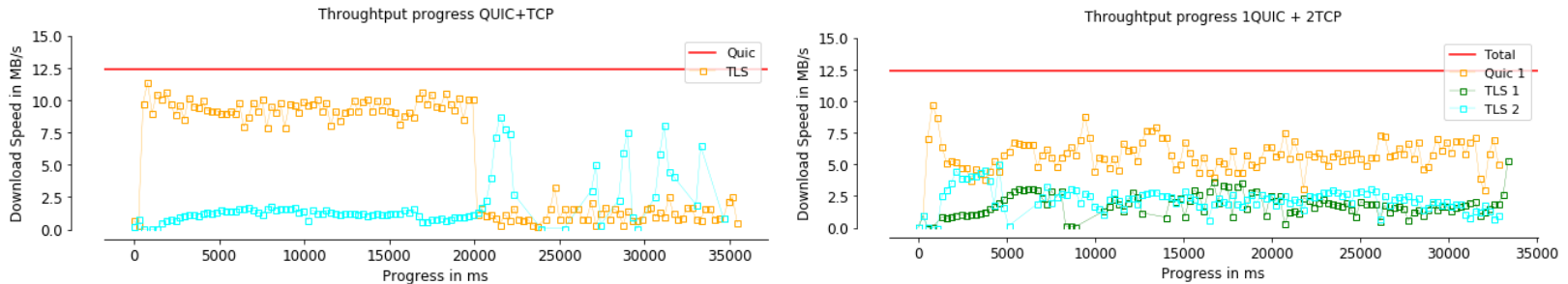


Figure: Throughput of 1 QUIC and 1 TCP, 1 QUIC and 2 TCP concurrent flow with the download progress of a 30 second YouTube video

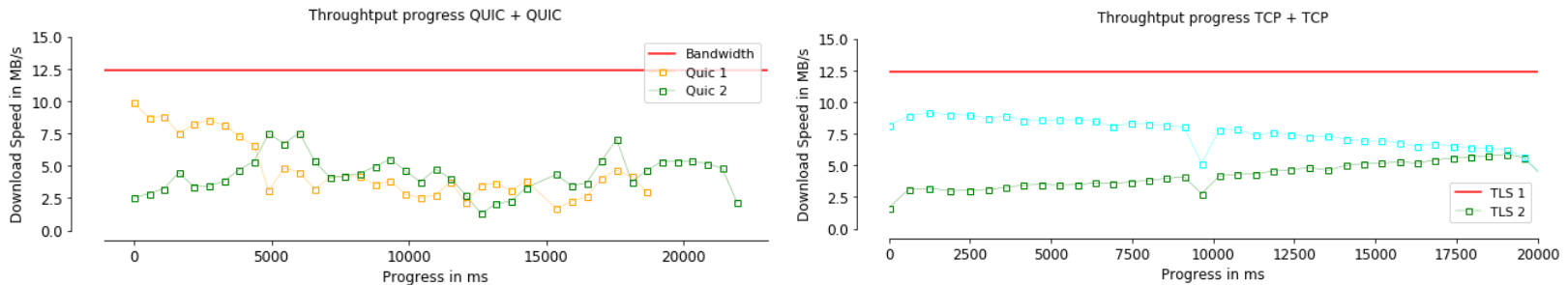


Figure: Throughput of 2 QUIC flows and 2 TCP flows with the download progress of a 200 MB file from Google Drive

Fairness

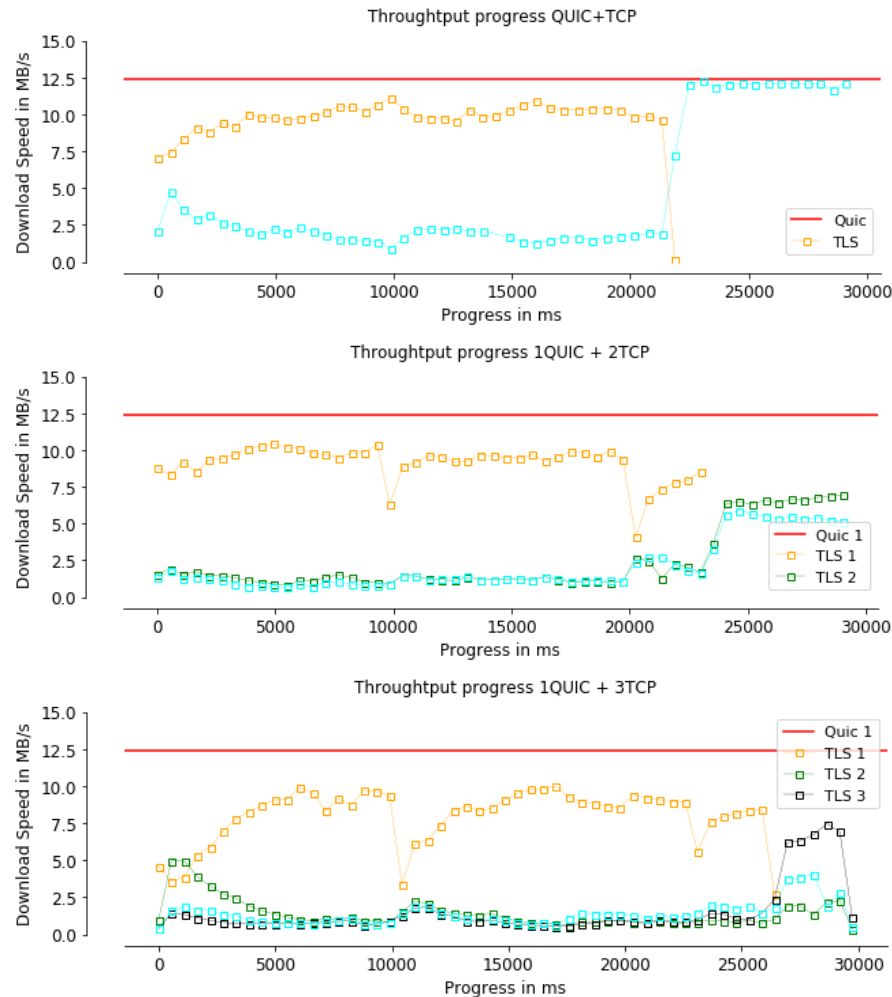


Figure: Throughput of 1 QUIC and Multiple TCP flow with the download progress of a 200 MB file from Google Drive

CPU Utilization

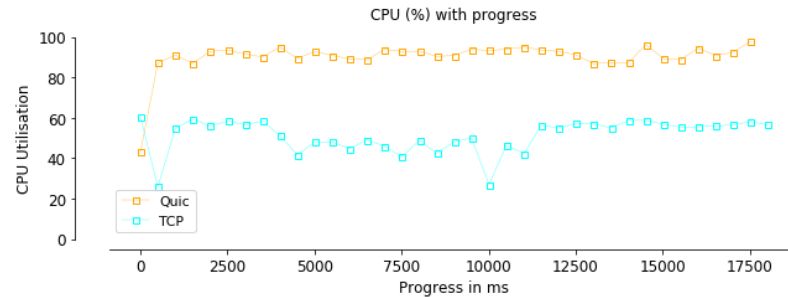


Figure: CPU utilization of QUIC(Q044) and TCP downloading a 200 MB file from Google Drive

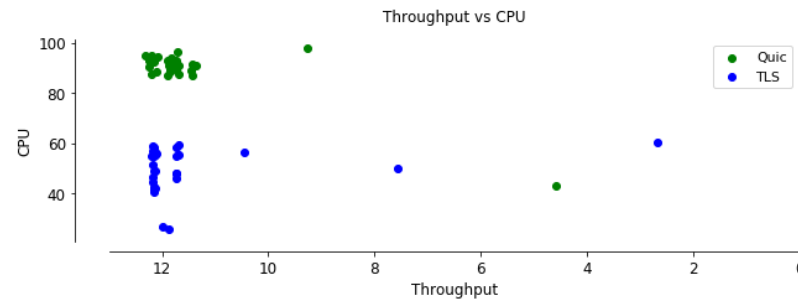


Figure: Scatter Plot for Throughput and CPU for QUIC and TCP while downloading from Google Drive

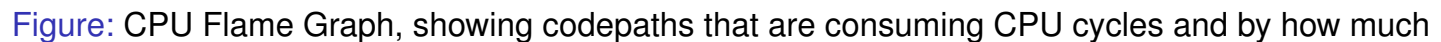
QUIC Profiling

| index | %time | seconds | usecs/call | calls | syscall |
|-------|-------|----------|------------|-------|------------|
| 0 | 76.89 | 1.949677 | 24 | 80800 | recvmsg |
| 1 | 17.33 | 0.439416 | 201 | 2191 | epoll_wait |
| 2 | 4.79 | 0.121456 | 57 | 2141 | sendmsg |
| 3 | 0.22 | 0.005475 | 57 | 96 | brk |
| 4 | 0.18 | 0.004439 | 2220 | 2 | poll |
| 5 | 0.08 | 0.00201 | 53 | 38 | open |
| 6 | 0.08 | 0.001954 | 43 | 45 | read |
| 7 | 0.07 | 0.001712 | 54 | 32 | mmap2 |
| 8 | 0.05 | 0.001261 | 53 | 24 | mprotect |
| 9 | 0.05 | 0.001185 | 41 | 29 | close |
| 10 | 0.03 | 0.000878 | 38 | 23 | fstat64 |

Table: Top-10 system calls for QUIC while downloading file from Google drive

QUIC uses a maximum packet size of 1350 for IPv4 and 1370 for IPv6 to avoid packet fragmentation.

Thus QUIC has to make many calls to send/recv system calls.



Rohit Panda (TUM) | rohit.panda@tum.de | Master's Thesis Presentation

Conclusion

- QUIC outperforms TLS in connection establishment times for both IPv4 and IPv6.
- From the measurements done in India we find greater improvements for QUIC over TCP/TLS than in Munich which shows that benefits of QUIC is more evident in low-bandwidth, high-loss networks and high-RTT regions.
- Q044 may have slightly better Connection times but TTFB and download time between different QUIC versions is similar for IPv4.
- Overall download rate in case of Video content delivery for TLS 1.2 is higher than QUIC.
- Q035 has higher download rates and lesser stall durations than other QUIC versions.
- Google AS serves an overwhelming majority of the websites using QUIC, almost half of the websites in our target list. Only 5 ASes account for 85% of our target websites.

Conclusion

- Mean throughput of QUIC is over twice that of TCP.
- QUIC performs better than TCP for smaller file sizes but for larger file sizes TCP has higher throughput.
- CPU utilization in case of QUIC is almost twice that of TCP
- QUIC spends most of its time in the kernel on expensive sendmsg/recvmmsg system calls.
- QUIC grabs more than its fair share of bandwidth when it is competing with TCP. QUIC flows are fair to other QUIC flows and TCP flows are fair to other TCP flows.

Future work and limitations

- For more representative results the tests should be deployed at multiple locations with the help of CAIDA Ark [4] and SamKnows project[30]
- Adopt tests need for mobile devices using cronet[6].
- No support for 0-RTT connections in Isquic at the time of test creation. Support has been introduced in latest release.
- The QUIC `YouTube tests` have a higher failure rate than TCP `YouTube tests`
- Other metrics like Congestion control window size and Packet pacing, no. of retransmissions, no. of packets lost.
- Tests under constrained conditions like low buffer size, introducing random bandwidth changes, random packet loss and different RTTs.

- Adopt the tests for IETF QUIC

Reproducibility

- The code is located in github repositories YouTube tests[36], quic_perf[26], tls_perf[32].
- The data and jupyter notebooks used for analysis are present on the VM *vmott16* and the github repository[18].
- Analysis for YouTube QoE has been done with scripts in [13]

References

- [1] A. Aggarwal, S. Savage, and T. E. Anderson. “Understanding the Performance of TCP Pacing”. In: *Proceedings IEEE INFOCOM 2000, The Conference on Computer Communications, Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, Reaching the Promised Land of Communications, Tel Aviv, Israel, March 26-30, 2000*. IEEE, 2000, pp. 1157–1165. DOI: 10.1109/INFCOM.2000.832483.
- [2] S. Ahsan, V. Bajpai, J. Ott, and J. Schönwälder. “Measuring YouTube from Dual-Stacked Hosts”. In: *Passive and Active Measurement - 16th International Conference, PAM 2015, New York, NY, USA, March 19-20, 2015, Proceedings*. Ed. by J. Mirkovic and Y. Liu. Vol. 8995. Lecture Notes in Computer Science. Springer, 2015, pp. 249–261. DOI: 10.1007/978-3-319-15509-8_19.
- [3] W. de Bruijn. “Optimizing UDP for content delivery: GSO, pacing and zerocopy”.
- [4] *CAIDA Ark*. URL: <http://www.caida.org/projects/ark/>.
- [5] S. Cook, B. Mathieu, P. Truong, and I. Hamchaoui. “QUIC: Better for what and for whom?” In: *IEEE International Conference on Communications, ICC 2017, Paris, France, May 21-25, 2017*. IEEE, 2017, pp. 1–6. DOI: 10.1109/ICC.2017.7997281.
- [6] *cronet*. URL: <https://developer.android.com/guide/topics/connectivity/cronet/>.
- [7] Y. Cui, T. Li, C. Liu, X. Wang, and M. Kuhlewind. “Innovating transport with QUIC: Design approaches and research challenges”. In: *IEEE Internet Computing 21.2* (2017), pp. 72–76. ISSN: 10897801. DOI: 10.1109/MIC.2017.44.
- [8] Y. Cui, T. Li, C. Liu, X. Wang, and M. Kühlewind. “Innovating Transport with QUIC: Design Approaches and Research Challenges”. In: *IEEE Internet Computing 21.2* (2017), pp. 72–76. DOI: 10.1109/MIC.2017.44.
- [9] K. Edeline, M. Kühlewind, B. Trammell, E. Aben, and B. Donnet. “Using UDP for Internet Transport Evolution”. In: *CoRR abs/1612.07816* (2016). arXiv: 1612.07816.

- [10] M. Fischlin and F. Günther. “Multi-Stage Key Exchange and the Case of Google’s QUIC Protocol”. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*. Ed. by G. Ahn, M. Yung, and N. Li. ACM, 2014, pp. 1193–1204. DOI: 10.1145/2660267.2660308.
- [11] A. Ford, C. Raiciu, M. J. Handley, and O. Bonaventure. *TCP Extensions for Multipath Operation with Multiple Addresses*. RFC 6824. Jan. 2013. DOI: 10.17487/RFC6824.
- [12] *github repository. The QUIC Tracker*. URL: <https://github.com/QUIC-Tracker/quic-tracker>.
- [13] *Gitlab repository. YouTube thesis documentation and plots scripts*. URL: <https://gitlab.lrz.de/cm/2018-sergey-masters-thesis>.
- [14] *Google design document. QUIC Crypto*. URL: https://docs.google.com/document/d/1g5nIXAIkN_Y-7XJW5K45Ib1Hd_L2f5LTaDUDwvZ5L6g/.
- [15] *Google design document. QUIC from 10,000 feet*. URL: <https://docs.google.com/document/d/1gY9-YNDNAB1eip-RTPbqphgySwSNSDHLq9D5Bty4FSU/>.
- [16] J. Iyengar and I. Swett. *QUIC Loss Detection and Congestion Control*. Internet-Draft draft-ietf-quic-recovery-18. Work in Progress. Internet Engineering Task Force, Jan. 2019. 36 pp.
- [17] B. Jaeger. “Measuring Google QUIC Connection Establishment Times”. Bachelor’s Thesis. Technische Universität München.
- [18] *Jupyter notebooks for analysis*. URL: <https://github.com/RohitPanda/master-thesis>.
- [19] A. M. Kakhki, S. Jero, D. R. Choffnes, C. Nita-Rotaru, and A. Mislove. “Taking a long look at QUIC: an approach for rigorous evaluation of rapidly evolving transport protocols”. In: *Proceedings of the 2017 Internet Measurement Conference, IMC 2017, London, United Kingdom, November 1-3, 2017*. Ed. by S. Uhlig and O. Maennel. ACM, 2017, pp. 290–303. DOI: 10.1145/3131365.3131368.

- [20] P. K. Kharat, A. Rege, A. Goel, and M. Kulkarni. “QUIC Protocol Performance in Wireless Networks”. In: *2018 International Conference on Communication and Signal Processing (ICCSP)*. Apr. 2018, pp. 0472–0476. DOI: 10.1109/ICCSP.2018.8524247.
- [21] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. R. Iyengar, J. Bailey, J. Dorfman, J. Roskind, J. Kulik, P. Westin, R. Tenneti, R. Shade, R. Hamilton, V. Vasiliev, W. Chang, and Z. Shi. “The QUIC Transport Protocol: Design and Internet-Scale Deployment”. In: *Proceedings of the Conference of the ACM Special Interest Group on Data Communication, SIGCOMM 2017, Los Angeles, CA, USA, August 21-25, 2017*. ACM, 2017, pp. 183–196. DOI: 10.1145/3098822.3098842.
- [22] F. Li, J. W. Chung, X. Jiang, and M. Claypool. “TCP CUBIC versus BBR on the Highway”. In: *Passive and Active Measurement - 19th International Conference, PAM 2018, Berlin, Germany, March 26-27, 2018, Proceedings*. Ed. by R. Beverly, G. Smaragdakis, and A. Feldmann. Vol. 10771. Lecture Notes in Computer Science. Springer, 2018, pp. 269–280. DOI: 10.1007/978-3-319-76481-8_20.
- [23] K. Nepomuceno, I. N. d. Oliveira, R. R. Aschoff, D. Bezerra, M. S. Ito, W. Melo, D. Sadok, and G. Szabó. “QUIC and TCP: A Performance Evaluation”. In: *2018 IEEE Symposium on Computers and Communications (ISCC)*. June 2018, pp. 00045–00051. DOI: 10.1109/ISCC.2018.8538687.
- [24] S. Podanev. “Measuring YouTube Video Content Delivery over QUIC Protocol”. Master’s Thesis. Technische Universität München.
- [25] P. Qian, N. Wang, and R. Tafazolli. “Achieving Robust Mobile Web Content Delivery Performance Based on Multiple Coordinated QUIC Connections”. In: *IEEE Access* 6 (2018), pp. 11313–11328. DOI: 10.1109/ACCESS.2018.2804222.
- [26] *quic_perf*. URL: <https://github.com/RohitPanda/lsqlquic-client>.

- [27] S. Radhakrishnan, Y. Cheng, J. Chu, A. Jain, and B. Raghavan. “TCP fast open”. In: *Proceedings of the 2011 Conference on Emerging Networking Experiments and Technologies, Co-NEXT '11, Tokyo, Japan, December 6-9, 2011*. Ed. by K. Cho and M. Crovella. ACM, 2011, p. 21. DOI: 10.1145/2079296.2079317.
- [28] *RIPE DB*. URL: <https://stat.ripe.net/>.
- [29] J. Rüth, I. Poese, C. Dietzel, and O. Hohlfeld. “A First Look at QUIC in the Wild”. In: *Passive and Active Measurement - 19th International Conference, PAM 2018, Berlin, Germany, March 26-27, 2018, Proceedings*. Ed. by R. Beverly, G. Smaragdakis, and A. Feldmann. Vol. 10771. Lecture Notes in Computer Science. Springer, 2018, pp. 255–268. DOI: 10.1007/978-3-319-76481-8_19.
- [30] *SamKnows project*. URL: <https://samknows.com/>.
- [31] A. Saverimoutou, B. Mathieu, and S. Vaton. “Which secure transport protocol for a reliable HTTP/2-based web service: TLS or QUIC?”. In: *2017 IEEE Symposium on Computers and Communications, ISCC 2017, Heraklion, Greece, July 3-6, 2017*. IEEE Computer Society, 2017, pp. 879–884. DOI: 10.1109/ISCC.2017.8024637.
- [32] *tls_perf*. URL: https://github.com/RohitPanda/tls_perf.
- [33] P. D. Vaere, T. Bühler, M. Kühlewind, and B. Trammell. “Three Bits Suffice: Explicit Support for Passive Measurement of Internet Latency in QUIC and TCP”. In: *Proceedings of the Internet Measurement Conference 2018, IMC 2018, Boston, MA, USA, October 31 - November 02, 2018*. ACM, 2018, pp. 22–28.
- [34] P. Wang, C. Bianco, J. Riihijärvi, and M. Petrova. “Implementation and Performance Evaluation of the QUIC Protocol in Linux Kernel”. In: *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM 2018, Montreal, QC, Canada, October 28 - November 02, 2018*. Ed. by A. Boukerche, S. S. Kanhere, and P. Bellavista. ACM, 2018, pp. 227–234. DOI: 10.1145/3242102.3242106.
- [35] J. Yoakum and L. Ong. *An Introduction to the Stream Control Transmission Protocol (SCTP)*. RFC 3286. May 2002. DOI: 10.17487/RFC3286.

- [36] *YouTube tests*. URL: <https://github.com/RohitPanda/Quic-Test>.
- [37] Y. Yu, M. Xu, and Y. Yang. “When QUIC meets TCP: An experimental study”. In: *36th IEEE International Performance Computing and Communications Conference, IPCCC 2017, San Diego, CA, USA, December 10-12, 2017*. IEEE, 2017, pp. 1–8. DOI: 10.1109/PCCC.2017.8280429.

