A Lightweight Performance Enhancing Proxy for Evolved Protocols and Networks

Zsolt Krämer*, Sándor Molnár*, Marcus Pieskä†, Attila Mihály‡
*Dept. of Telecomm. and Media Informatics, Budapest University of Technology and Economics, Hungary
{kramer, molnar}@tmit.bme.hu
†Dept. of Mathematics and Computer Science, Karlstad University, Sweden
marcus.pieska@kau.se
‡Traffic Analysis and Network Performance Laboratory, Ericsson Research, Hungary
attila.mihaly@ericsson.com

***Abstract*—In this paper we present the design and prototype implementation of a multi-domain congestion control framework based on a non-ossifying Lightweight Performance Enhancing Proxy. We demonstrate that our solution can be both used for evolved protocols such as end-to-end encrypted QUIC and for evolved cellular access networks like 5G. We also show the performance of our Multi-Domain Congestion Control algorithm by a simulation study. Moreover, we expose its behavior in realistic use cases like sudden capacity changes in 5G mmWave cellular networks. Our results highlight that significant performance improvements can be achieved by cooperative traffic management frameworks in protocols and networks of the future.**

***Index Terms*—QUIC, PEP, Network Management, Congestion Control, 5G**

I. Introduction

One of the main obstacles of the evolution of transport protocols is the ossification of the transport layer. It has long been known that TCP is very hard to extend [1], and a recent, large-scale measurement study [2] has revealed the prevalence of TCP/IP middleboxes. It has also pointed out the significant share of harmful ones among them. Google's QUIC (Quick UDP Internet Connections) protocol aims to overcome this by encrypting the transport headers resulting in immuninty to middlebox interference. As a regrettable side effect, this will make all potentially useful middlebox interactions (e.g., for performance improvement) impossible.

The new radio access technologies introduced in 5G enable extremely high data rates and low delays, however, the available capacity is very sensitive to line-of-sight (LOS) and non-line-of-sight (NLOS) transitions. To take advantage of the available capacity in 5G mmWave network, fast adaptation to the available bandwidth is going to be a crucial property of congestion control (CC) algorithms. It has been shown [3] that there is a clear general tradeoff between the above property and TCP fairness in CC algorithms. However, [3] assumes end-to-end congestion control, but if the sender can utilize additional information from the network and the CC can adapt its behavior to the location of the bottleneck, it is possible to provide both fast adaptation and TCP fairness.

The results presented in this paper are an outcome of our research carried out in recent years. We have presented the concept of a non-ossifying Lightweight Performance Enhancing Proxy (LwPEP) in [4] followed by the first design concepts and initial performance evaluation results of our Multi-Domain Congestion Control (MD CC) algorithm in [5] with its LTE simulation study in [6]. By utilizing the feedback from the LwPEP, MD CC has been shown to deliver significant performance benefits in LTE networks, while also being able to preserve TCP fairness.

In this paper, we present the prototype implementation of our Lightweight Performance Enhancing Proxy, which provides cooperative management support for both TCP and QUIC traffic. We show that the PEP handles encrypted traffic without privacy violation adding minimal delay to the end-to-end communication. We also discuss the important use-cases of Multi-Domain Congestion Control and bottleneck detection. In addition, we present interesting simulation results of our framework analyzing the performance in 5G mmWave networks.

The paper is organized as follows. Section II summarizes the related work on TCP performance in 5G mmWave cellular networks, multi-domain congestion control and QUIC manageability. The design and implementation of the Lightweight PEP prototype is presented in Section III, and the two main usecases are described in Section IV. After that, in Section V we illustrate the performance benefits of MD CC in mmWave simulations, and Section VI concludes the paper.

II. Related work

A. TCP performance in 5G mmWave networks

A wide range of questions regarding the interplay between transport layer mechanisms and 5G environments were explored by the authors of [7]. One key simulation result showed that Controlled Delay (CoDel) active queue management (AQM) is a viable technique to achieve low latency in 5G, however, with loss-based CC algorithms, the throughput is reduced compared to using an adequately sized droptail buffer. The paper also shows a significant difference in loss-based TCP performance between edge - and remote servers, which means that TCP benefits greatly from a short control loop in such dynamic radio conditions, and prompts the question of the role of TCP-splitting in 5G. The tradeoff between latency

and throughput was also studied by the authors of [8], where they have shown that it is in general difficult for TCP to deliver *both* high throughput *and* low latency when the link capacity varies as much as a mmWave link may vary between LOS and NLOS states [8].

A comprehensive performance evaluation of CC algorithms in 5G mmWave simulations was presented in [9]. The paper claims that a 7MB Radio Link Control (RLC) buffer achieves maximum throughput and mitigates the bufferbloat effect. The results show that while Scalable TCP [10] and CUBIC [11] can optimally recover after short NLOS periods, when high channel quality degradation is considered after a longer NLOS period, CUBIC attempts to recover from slow start and the success is highly variable.

B. Performance Enhancing Proxies and their alternatives

The prevalence of TCP splitting PEPs were explored in [12], where the authors showed that 86% of studied cellular networks employed such proxies. The authors of [13] used an LTE testbed and emulated mmWave-like bandwidth fluctuations to show the potential performance gains of TCP splitting PEPs in 5G mmWave networks.

Both [14] and [15] are targeting cooperative performance enhancement in 5G networks. In [14] the authors propose *Milliproxy*, an entity that modifies the advertised window in the acknowledgements sent by the client based on a flow window policy and relays them back to the server. The effective congestion window is determined as the minimum of the advertised window and the congestion window. An enhanced transport solution was designed specifically for Edge Cloud scenarios in [15]. The initial window is carefully inflated based on the state of the network buffer, provided by a Traffic Probe and a Traffic Control Function entity. The solution is based on the observation that in Edge Cloud scenarios, the radio access network is responsible for the resource sharing instead of the transport layer congestion control algorithm on the whole path. Both proposals showed promising performance results in simulations, however, both have open deployment questions.

ABC (Accel-Brake Control) [16] is a novel, cooperative congestion control mechanism for wireless networks. The ABC router computes a target rate and echoes a 1-bit signal (either accelerate or brake) to the sender through the client. Each ACK can instruct the sender to increase or decrease the congestion window by one packet, and thus, ABC is exceptionally scalable. Deployability, however, could be a serious concern as ABC signals are implemented by reinterpreting the ECN bits (using ECT(1) as accelerate and ECT(0) as brake). There are currently two proposals in the networking community for enhanced congestion signaling; SCE (Some Congestion Experienced) [17] and L4S (Low Latency, Low Loss, and Scalable throughput) [18] both of which use the ECT code points differently, and thus both would most likely be incompatible with ABC.

C. The QUIC protocol

The QUIC protocol successfully aimed at decreasing Internet latency by introducing a shorter handshake procedure (one - or even zero round-trip times), and multiplexing in the transport layer. However, QUIC is also designed from the ground up to minimize the risk of potential pathological behavior caused by middleboxes, and it encrypts most of the transport header. This presents serious challenges for network operators interested in managing QUIC traffic, as the built-in encryption of QUIC imposes limits to the capabilities of on-path management solutions.

QUIC supports passive round-trip time (RTT) measurement for middleboxes via an unencrypted spin bit in the header. The QUIC client sends packets with the same spin bit as the last received value from the server, while the server changes the last received value, and thus the on-path middleboxes can infer the RTT from the observed changes in the spin bit. An enhancement was presented in [19], where two additional bits are used for validation. There are recent ideas (e.g., [20], [21]) which would allow explicit cooperation between QUIC endpoints and middleboxes, however, these concepts are still very much in development. There are no published transparent solutions that could replace traditional Performance Enhancing Proxies for encrypted traffic.

III. Lightweight PEP prototype: Design and implementation

A. Acknowledging encrypted traffic without privacy violation

In [4] we have presented our concept for a PEP that sends safe-to-ignore, incrementally useful PEP-ACKs to TCP servers. In case of TCP, these ACKs would contain the sequence numbers seen by the proxy, as shown in the upper figure of Fig. 1 and can be used by the server as an input for enhanced congestion control schemes. However, if the transport headers are encrypted (as in QUIC) we need a different solution for acknowledging data seen by the PEP. Our solution to this problem is that, instead of sequence numbers, some parts of the original server message are sent back by the PEP. The amount of data to be sent back is chosen in a way that it also contains some part of the payload of the original server message even if long headers are used. The server needs to maintain a mapping between the sequence numbers and the payload slices, and thus, when it receives a PEP-ACK, the server can determine the sequence number from the encrypted payload slice sent back by the LwPEP. This also enables the

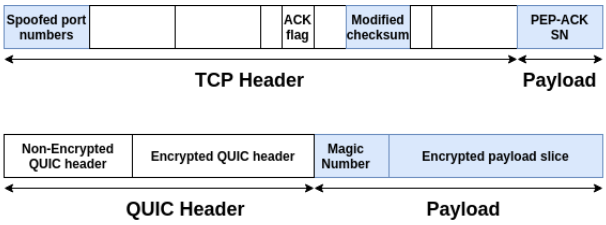

Fig. 1. Wire format of the PEP-ACKs for TCP and QUIC

server to verify that the PEP has seen the given packet. It is useful that the PEP includes a magic number in the ACK that helps identifying the PEP-ACKs on the server side. This results in the wire format shown in the lower figure of Fig. 1, used in the prototype implementation. Besides, the PEP could also send a digital signature that can identify the PEP. The PEP should also send its public key certificate, which is sufficient to be sent in the first few messages.

B. Implementation

The Lightweight PEP prototype was implemented as a userspace application using the netfilter_queue [22] library in Linux, which has proven to be efficient for userspace proxy applications in previous works [23]. The block diagram of the proxy can be seen in Figure 2. An iptables policy selects the packets to be sent to the queue, which in the QUIC case means filtering UDP packets on port 443. The packet number field in the QUIC header is of variable length, which is given by the encrypted packet number length field. This means that when parsing the QUIC packets, the proxy has to assume the maximum of the potential length of the header so that the size of the payload slice sent back to the server is 10 bytes. After parsing the filtered packets, the proxy immediately forwards them to the client with negligible added delay to the end-to-end communication. After the forwarding is complete, the LwPEP starts assembling the PEP-ACK, which is then sent back to the server via a RAW socket.

To illustrate the performance of the prototype in terms of delay, we have set up a mininet testbed with simple topology consisting of a server, a client and the LwPEP deployed between them. We used iperf to generate TCP traffic, and a LiteSpeed web server [24] with the IETF QUIC implementation, which powers 97% of the QUIC-enabled websites worldwide.

Table I shows an illustration of processing performance of the userspace LwPEP prototype. The LwPEP calculates a running average of both the time it takes to parse the incoming packets and forward them (Time to Forward) and the time it takes to assemble the PEP-ACKs and send them back through the RAW socket (Time to Ack). It can be seen that on average, the LwPEP adds a 0.06 ms delay to the end-to-end connection for TCP and 0.2 milliseconds for QUIC. Generating the PEP-ACKs takes an additional 0.012 ms for TCP on average and 0.495 ms for QUIC. These processing times are slightly larger for QUIC due to the complexity of the protocol. The userspace implementation provides flexibility for prototyping, however, note that the delays could be further decreased by an implementation in hardware, e.g., using P4.

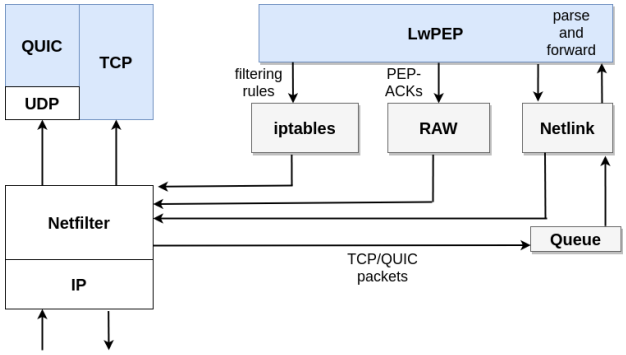

Fig. 2. Block diagram of the Lightweight PEP prototype

TABLE I
AVERAGE TIME OF FORWARDING AND PEP-ACK GENERATION FOR TCP AND QUIC

Protocol	Time to Forward [ms]			Time to Ack [ms]		
	Min	Average	Max	Min	Average	Max
TCP	0.055	0.060	0.531	0.063	0.072	0.806
QUIC	0.053	0.200	0.518	0.156	0.695	1.295

IV. USE CASES

A. Multi-Domain Congestion Control

With the LwPEP placed on the border of the wired- and cellular domains, the server can utilize the feedback from the proxy to optimize congestion control. Providing fairness is not the responsibility of the congestion control algorithm in the cellular access domain, however, maintaining traditional TCP fairness in the wired domain is still important when the bottleneck is located there.

By utilizing both the PEP-ACKs and the regular end-to-end ACKs, it is possible to design a Multi-Domain CC scheme as we have shown it in [5]. MD CC runs two internal congestion control algorithms in parallel. The PEP-ACKs are clocking a CUBIC algorithm, while the client ACKs are clocking a Scalable TCP algorithm with increased aggressiveness. The effective congestion window is always set to the minimum of the two components and thus a multi-domain CC algorithm built on the LwPEP feedback can effectively adapt to the location of the bottleneck.

In [6] we have shown that in LTE networks, MD CC achieved a 7.3% average improvement in long term throughput over CUBIC. Also, in cases of sudden increases in available capacity, MD CC was able to utilize the new bandwidth faster. We have also shown that Multi-Domain CC achieves a significant reduction in short flow completion times compared to CUBIC. Section V of this paper shows simulation results on how the Multi-Domain CC can achieve better utilization than CUBIC in 5G mmWave networks.

B. Bottleneck detection

Detecting whether the bottleneck is in the wired - or the cellular domain can benefit servers and server-side applications in multiple ways. One example is when a server detects that the bottleneck is not in the cellular domain, then the traffic could potentially be routed to the destination by avoiding this bottleneck. Bottleneck detection is an intensively studied problem with mature and efficient proposals, however, we argue that our framework presents a unique solution with passive, vendor-agnostic server-side bottleneck detection.

A passive, client-side technique was presented for LTE networks in [25], where the UE is able to differentiate between cellular - and wired bottlenecks based on the observed allocation pattern in a bandwidth exploration algorithm. This bottleneck information is utilized in the calculation of an optimal congestion window which is then relayed to the server. The framework is able to quickly detect the bottleneck (in a few RTTs), however the explicit bottleneck location is not available to the server. BurstTracker [26] is another client-side algorithm, and it works by exploiting observations on the behavior of the downlink scheduling algorithms in LTE networks, which reveal the status of downlink queues. The authors state that while the core principle of the solution is future proof, the algorithm built on it is LTE specific and thus it is likely not suitable for 5G cellular networks.

QProbe [27] is an active approach, meaning that it sends probing traffic from the server, and observes the arrival times at the client. The bottleneck detection algorithm is based on the the different behavior of the FIFO queues in the wired domain and the proportional fair schedulers in the cellular domain. The authors show that QProbe is able to differentiate between WAN and cellular bottlenecks in time the order of 700 ms. The paper also presents a comprehensive measurement study across numerous operators and countries, where the bottleneck was found to be in the cellular domain for 68.9% and 25.7% of the cases for 3G and LTE, respectively.

The implicit bottleneck detection used in Multi-Domain Congestion Control can be provided for servers as an explicit bottleneck detection solution in addition to the benefits of running the advanced CC. Built on the acknowledgements from the Lightweight PEP, MD CC can provide bottleneck detection for servers that is:

- passive,
- vendor-agnostic and
- non-ossifying.

V. Performance in 5G mmWave networks

A. Simulation environment

The simulations were carried out using the mmWave module [28] developed for the ns-3 network simulator. The Direct Code Execution (DCE) cradle [29] was also used, extended with our MD CC implementation in the linux kernel (version 4.7.0. which is the latest available version in the NUSE [30] library operating system's network stack). Figure 3 depicts the topology used. We have tested varying RTTs in the Internet-domain between 2 and 40 ms. The RLC was used in Acknowledged Mode, with different (fixed) buffer sizes and a CoDel AQM. The LwPEP is co-located with the PGW. Table II enumerate values used for various parameters in the topology. The physical environment and mobility is accounted for in coordinates, with the user moving steadily between its start and stop way-point; the rectangle obstacle is defined by two opposite corners.

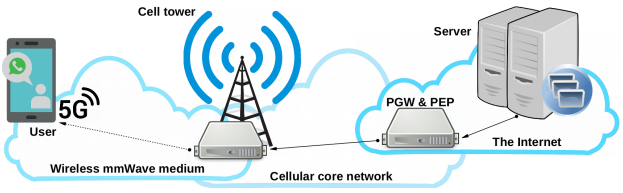

Fig. 3. Topology used in the simulations.

TABLE II
Configuration of ns3.

Parameter	ns-3 value
Internet link RTT	{2 - 40} ms
Internet link bandwidth	100 Gbps
Core network RTT	2 ms
Coordinates of eNodeB	(0, 0, 30) m
User start point	(50, 15, 1.5) m
User end point	(50, -15, 1.5) m
Obstacle corner 1	(40, -4, 0) m
Obstacle corner 2	(45, 4, 30) m
MAC Scheduler	MmWaveFlexTtiMacScheduler
Pathloss model	BuildingsObstaclePropagationLossModel
Carrier frequency	28 GHz
RLC mode	AM
RLC buffer size	{2,7,20} MB
Chunk per RBs	72
CoDel target delay	5 ms
Traffic generator	Iperf

B. Performance results

We have simulated 15 seconds of bulk TCP transmission using iperf, with an approximately 5 seconds long NLOS period in between periods of LOS. The bandwith of the Internet link was set to 100 Gbps in order to simulate a wireless bottleneck in both periods. The transition to NLOS decreases the available capacity significantly, and as can be seen in Figure 4, TCP throughput is also decreased for both MD CC and CUBIC. The extent of this decrease however is very different. The throughput results for three different RLC buffer sizes are depicted for the duration of the whole transmission. In the NLOS period, it can be seen that CUBIC can not utilize the available capacity if the link is underbuffered (2MB). Smaller buffers benefit MD CC's performance compared to CUBIC in the LOS periods as well. With a 2MB buffer, CUBIC is not able to utilize the available capacity even before the NLOS transition. A large buffer (20MB) can improve CUBIC's recovery after the NLOS period at the cost of significantly increased delay. The throughput of the MD CC flow is relatively insensitive to the buffer size, mainly because its resilience against losses due to the aggressiveness of the end-to-end component.

The effect of the RLC buffer size on the performance benefits of MD CC is further studied in Table III. The table shows that MD CC achieves nearly the same average throughput in the 2-20 MB range. The only configuration when the performance of CUBIC is comparable is the one with a large buffer, where the difference in average throughput is 3.5%. If deep buffers are avoided in order to achieve lower latency, MD CC with PEP feedback outperforms CUBIC, with

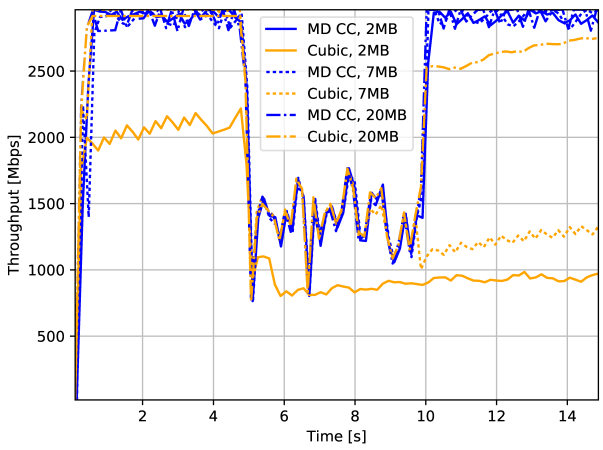

Fig. 4. Throughput of MD CC and CUBIC with different buffer sizes

TABLE III
AVERAGE THROUGHPUT AND RTT WITH DIFFERENT RLC BUFFER SIZES

RLC buffer size	Avg throughput [Mbps]		Avg RTT [ms]	
	CUBIC	Multi-Domain CC	CUBIC	Multi-Domain CC
2 MB	1256	2348	22	26
7 MB	1811	2358	31	44
20 MB	2283	2366	48	87

the performance gains ranging from a 23.2% in the 7MB case to 46.5% with a 2MB buffer. The average smoothed RTT seen by the transport shows an interesting trend, as for the same buffer size, CUBIC achieves lower latency, which is expected as MD CC fills the buffer aggressively, however, if we compare the latency in the cases with similar average throughput, the latency is 45.8% lower with MD CC and a 2MB buffer than with CUBIC and a 20MB buffer.

Next, we studied the performance benefits of MD CC with different RTTs on the Internet leg. Figure 5 summarizes our findings in the 2-40ms RTT range for both a 7MB droptail buffer and a CoDel AQM used in the RLC. CUBIC achieves comparable average throughput only in the 2ms RTT

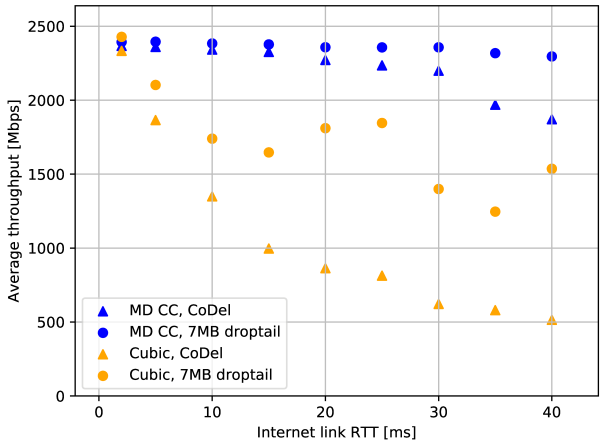

Fig. 5. Average throughput of MD CC and CUBIC flows with different RTTs

case, which corresponds to an edge computing scenario with servers placed very close to the user. As the Internet link RTT increases, the average throughput of MD CC is only slightly affected compared to the steep decrease in CUBIC's throughput. Both algorithms perform better with a 7MB fixed buffer, where MD CC consistently outperforms CUBIC with a difference in average throughput between 13.9% and 86%. When CoDel is used, MD CC achieves higher throughput than CUBIC by 73.5%, 163.1% and 263.9% in the cases of 10, 20, and 40 ms Internet link RTT, respectively.

VI. CONCLUSION

In this paper, we have presented the design and implementation of a non-ossifying Lightweight Performance Enhancing Proxy, to support cooperative management of both TCP and QUIC traffic. By sending back parts of the encrypted payload, the PEP is able to acknowledge encrypted traffic without privacy violation. Based on our measurements we demonstrate that the PEP adds only a minimal delay to the end-to-end communication resulting in minimal impact on the end-to-end performance because of adding PEP on path.

One key use case of the feedback provided by the proxy is Multi-Domain Congestion Control (MD CC) that has been proven to adapt more rapidly to cellular conditions already for LTE so the expectation was that this is even more pronounced for the higher-bandwidth, but more volatile 5G access. We have shown in mmWave simulations considering typical LOS-NLOS transitions that MD CC significantly outperforms CUBIC in a wide range of different RLC buffer configurations and Internet-domain round-trip times.

The above results show that the Lightweight PEP concept could be a powerful tool to include in cooperative traffic management frameworks in order to enhance the performance not only in current networking scenarios but also in the future ones with evolved transport protocols and cellular access networks. Our future work therefore focuses on understanding the ecosystem that enables cooperating with such a PEP.

ACKNOWLEDGEMENT

The authors would like to thank Péter Takács for his contributions.

REFERENCES

[1] M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, and H. Tokuda, "Is it still possible to extend tcp?" in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, 2011, pp. 181–194.

[2] K. Edeline and B. Donnet, "A bottom-up investigation of the transport-layer ossification," in *2019 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 2019, pp. 169–176.

[3] D. Zarchy, R. Mittal, M. Schapira, and S. Shenker, "Axiomatizing congestion control," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 3, no. 2, pp. 1–33, 2019.

[4] A. Mihály, S. Nádas, S. Molnár, Z. Krämer, R. Skog, and M. Ihlar, "Supporting multi-domain congestion control by a lightweight pep," in *Internet of Things, Embedded Systems and Communications (IINTEC), 2017 International Conference on*. IEEE, 2017, pp. 105–110.

[5] Z. Krämer, S. Molnár, A. Mihály, and S. Nádas, "Towards multi-domain congestion control in next-generation networks," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2019, pp. 1–7.

[6] Z. Krämer, S. Molnár, S. Solymos, and A. Mihály, "On the benefits of multi-domain congestion control in lte networks," in *2019 IEEE 24th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. IEEE, 2019, pp. 1–6.

[7] M. Zhang, M. Polese, M. Mezzavilla, J. Zhu, S. Rangan, S. Panwar, and M. Zorzi, "Will tcp work in mmwave 5g cellular networks?" *IEEE Communications Magazine*, vol. 57, no. 1, pp. 65–71, 2019.

[8] M. Pieska and A. Kassler, "Tcp performance over 5g mmwave links—tradeoff between capacity and latency," in *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2017, pp. 385–394.

[9] P. Jimenez Mateo, C. Fiandrino, and J. Widmer, "Analysis of tcp performance in 5g mm-wave mobile networks," in *Communications (ICC), 2019 IEEE International Conference on*, 2019.

[10] T. Kelly, "Scalable tcp: Improving performance in highspeed wide area networks," *ACM SIGCOMM computer communication Review*, vol. 33, no. 2, pp. 83–91, 2003.

[11] S. Ha, I. Rhee, and L. Xu, "Cubic: a new tcp-friendly high-speed tcp variant," *ACM SIGOPS operating systems review*, vol. 42, no. 5, pp. 64–74, 2008.

[12] R. Zullo, A. Pescapé, K. Edeline, and B. Donnet, "Hic sunt proxies: Unveiling proxy phenomena in mobile networks," in *2019 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 2019, pp. 227–232.

[13] D. A. Hayes, D. Ros, and O. Alay, "On the importance of tcp splitting proxies for future 5g mmwave communications."

[14] M. Polese, M. Mezzavilla, M. Zhang, J. Zhu, S. Rangan, S. Panwar, and M. Zorzi, "milliproxy: a tcp proxy architecture for 5g mmwave cellular systems," *arXiv preprint arXiv:1712.02700*, 2017.

[15] Å. Arvidsson and L. Westberg, "Fast transport for edge computing in 5g networks," in *2018 26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, 2018, pp. 1–5.

[16] P. Goyal, A. Agarwal, R. Netravali, M. Alizadeh, and H. Balakrishnan, "Abc: A simple explicit congestion control protocol for wireless networks," *arXiv preprint arXiv:1905.03429*, 2019.

[17] J. Morton and D. M. Täht, "The Some Congestion Experienced ECN Codepoint," Internet Engineering Task Force, Internet-Draft draft-morton-taht-sce-00, Mar. 2019, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-morton-taht-sce-00

[18] B. Briscoe, K. D. Schepper, M. Bagnulo, and G. White, "Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture," Internet Engineering Task Force, Internet-Draft draft-ietf-tsvwg-l4s-arch-06, Mar. 2020, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-tsvwg-l4s-arch-06

[19] P. De 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*, 2018, pp. 22–28.

[20] M. Kühlewind and B. Trammell, "Manageability of the QUIC Transport Protocol," Internet Engineering Task Force, Internet-Draft draft-ietf-quic-manageability-06, Jan. 2020, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-quic-manageability-06

[21] M. Kühlewind, Z. Sarker, T. Fossati, and L. Pardue, "Use Cases and Requirements for QUIC as a Substrate," Internet Engineering Task Force, Internet-Draft draft-kuehlewind-masque-quic-substrate-00, Mar. 2020, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-kuehlewind-masque-quic-substrate-00

[22] "libnetfilter_queue project." [Online]. Available: https://www.netfilter.org/projects/libnetfilter_queue/

[23] G. Hampel, A. Rana, and T. Klein, "Seamless tcp mobility using lightweight mptcp proxy," in *Proceedings of the 11th ACM international symposium on Mobility management and wireless access*, 2013, pp. 139–146.

[24] "Litespeed quic library." [Online]. Available: https://github.com/litespeedtech/lsquic

[25] X. Xie, X. Zhang, and S. Zhu, "Accelerating mobile web loading using cellular link information," in *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, 2017, pp. 427–439.

[26] A. Balasingam, M. Bansal, R. Misra, K. Nagaraj, R. Tandra, S. Katti, and A. Schulman, "Detecting if lte is the bottleneck with bursttracker," in *The 25th Annual International Conference on Mobile Computing and Networking*, 2019, pp. 1–15.

[27] N. Baranasuriya, V. Navda, V. N. Padmanabhan, and S. Gilbert, "Qprobe: locating the bottleneck in cellular communication," in *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, 2015, pp. 1–7.

[28] M. Mezzavilla, M. Zhang, M. Polese, R. Ford, S. Dutta, S. Rangan, and M. Zorzi, "End-to-end simulation of 5g mmwave networks," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2237–2263, 2018.

[29] H. Tazaki, F. Uarbani, E. Mancini, M. Lacage, D. Camara, T. Turletti, and W. Dabbous, "Direct code execution: Revisiting library os architecture for reproducible network experiments," in *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*. ACM, 2013, pp. 217–228.

[30] H. Tazaki, R. Nakamura, and Y. Sekiya, "Library operating system with mainline linux network stack," *Proceedings of netdev*, 2015.