Segmenting Latency in a Private 4G LTE Network

Sophie Smith Ishan Darwhekar James Blakley
Thomas Eiszler Jan Harkes

May 2022
CMU-CS-22-115

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Many edge-native applications require low and predictable end-to-end network latency. In practice, many user-interactive edge applications must deliver less than 50ms round trip times (RTT) from the client device to an edge cloudlet and back to the client device to achieve acceptable user experience. More intensive interactive applications like virtual reality require less than 20ms RTTs. However, commercial 4G LTE networks fail to reliably meet these thresholds. This report summarizes the results of our measurement of the sources of network latency in an operational private outdoor CBRS 4G LTE network to provide a baseline for future network latency optimization.

This research was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0051 and by the National Science Foundation (NSF) under grant number CNS-1518865 and the NSF Graduate Research Fellowship under grant numbers DGE1252522 and DGE1745016. Additional support was provided by Intel, Vodafone, Deutsche Telekom, Crown Castle, Amazon Web Services, JMA Wireless, InterDigital, Seagate, Microsoft, VMware and the Conklin Kistler family fund. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view(s) of their employers or the above funding sources.

Keywords: edge computing, mobile networks

1 Introduction

Pervasive computing applications like augmented reality have experienced recent steady growth. User mobility is a pivotal requirement in these use cases. Mobility requires reliable and agile wireless connectivity. Such applications are typically computationally expensive and need the support of edge infrastructure to offload some time-sensitive processing to the edge. This offloading is beneficial to the mobile devices in two ways. First, device battery consumption is reduced, and, second, a cloudlet on the network edge has more computation capacity to deliver the results faster. This support, however, requires a near real-time response in communication between the device and edge.

In this regard, multiple wireless technologies have been developed for varying usage scenarios. Wi-Fi (IEEE 802.11) has emerged as the dominant technology for wireless connectivity indoors; cellular technologies like LTE (4G) and 5G are primarily used outdoors. Networks deployed outdoors, such as LTE, suffer from more unpredictability compared to indoor networks, such as Wi-Fi. Maintaining connectivity during mobility, supporting 500 to 1000 meter connectivity, and changing environmental conditions are some factors that add to this complexity. Additionally, LTE suffers from delay in communication during the wireless access and the back-end processing. Its successor technology, 5G, is being developed and deployed currently, so it is essential to understand the sources of delay in the predominant LTE technology to better inform 5G implementation. To outline why these network delays are introduced, a short background in LTE is presented below. The general architecture of the private LTE network used in this work is shown in Figure 1 and is based on the 3GPP specification [1].

An LTE network can be broadly divided into two sections - the Radio Access Network (RAN) and the Core Network (CN). The RAN includes the wireless access equipment used to communicate with the mobile device and referred to as the base station or eNodeB. The eNodeB consists of a field deployed radio unit and a backend processing unit. The eNodeB is connected to the CN by the backend processing unit. The CN is primarily composed of the Evolved Packet Core (EPC) and the Packet Data Network Gateway (PGW). The EPC is connected with multiple eNodeBs on one end and to the PGW on the other. PGW is the connection point of the LTE network with the external IP network and connects the User Equipment (UE) to the Internet. Each of these components, along with the UE, adds to the round-trip latency. The latency includes component level processing delays and an unavoidable transportation delay between components. Thus, understanding which network segments add the most incremental latency to the end-to-end latency will help in the optimization of both LTE and 5G networks.

This technical report presents our findings from the latency experimentation and segmentation of a field-deployed private LTE network with use of Commercial off-the-shelf (COTS) UEs for measurement. Wireshark [2] was used to place multiple probes in the network to capture relevant packet information. Extensive studies and their results are presented under various test conditions to realize their effects on latency. The rest of the report is structured as follows - Section 2 highlights the related work on measurement of network latency in LTE; Section 3 explains the private LTE setup used for experimentation; Section 4 describes the framework used for segmenting the network; Section 5 presents a detailed analysis of the experiments and their observations; Section 6 points towards directions for optimizing the network for lower latency; lastly, the conclusions, acknowledgements and the appendix are shared in Section 7, 8, and 9 respectively.

2 Related Work

Latency in communication is a critical factor in network performance. Understanding the latency in transport of packets from the source to the destination is paramount when time-sensitive applications are in use. Round-trip latency is defined as the difference in time from the transmission of the packet at the transmitter to the reception of a response to this packet back at the original transmitter. There have been multiple studies and significant documentation exists on the topic of round-trip latency in LTE networks. Enzo [3] presents a simulation based study on OPNET with an LTE network designed to emulate the real world scenario. The authors calculated a round-trip latency of 81 ms at a distance of 500 m and a latency of 97 ms at a distance of 1000 m at application level between user and the base station. Green et. al. in [4] carried out a field trial for measuring the performance of a LTE FDD (Frequency Division Duplexing) network and observed delays of 24 ms and 32 ms for users close by and far from the eNodeB respectively.

Multiple studies in the literature present the latency measurements for particular applications to ensure the necessary Quality of Service (QoS) expected by the user. A latency analysis is presented for vehicular applications in [5], concluding that the field deployed network yields a latency below 100 ms, which is acceptable as per the standards prescribed for vehicular communication. Xu [6] presents the case in for Smart Grids and illustrates the increase in latency with packet size. Maskey considers Machine to Machine (MTM) applications in LTE networks in [7], and obtained 29.32 ms as the average round-trip latency between GPS clock synchronized devices.

Our own work at the Living Edge Lab [8], [9], [10], [11], [12] has focused on empirically measuring round-trip latency in edge and cloud computing environments and understanding the implication for specific edge-native applications.

In the above studies, good signal and connectivity conditions are considered. However, there are multiple scenarios when the latency experienced will be worse than normal operating conditions. These overheads are incurred due to procedures in LTE meant to either establish a connection at the start or maintain the connection due to mobility. There could be external factors like signal quality and network congestion additionally impacting the latency. Maskey [7] presents a measurement that the Radio Resource Control (RRC) which manages signalling between UE and eNodeB at Layer 3, requires 86.8 ms on average in the connection setup phase. Since support for mobility is a major aspect of LTE networks, a handover, which is the transfer of connectivity of a UE from one eNodeB to another is important. An additional signalling overhead of 15.6 ms is incurred during handover as per the studies presented by [13]. Mesbahi [14] shares a study of delay and jitter in LTE networks concluding that the requirements in Quality of Service (QoS) depend on the geographical size supported by a base station radio and the network configuration at the eNodeB. The cellular operators need to ensure each network element is optimized to guarantee the QoS.

Despite such voluminous literature on round-trip latency measurements, there is a dearth of studies on segmentation of the network latency in LTE. Segmentation of network latency is meant to understand the delay introduced by each component of the network and realize the primary contributors to the round-trip latency. Segment latency is the time for a packet to transit through a specific network segment. Segments can be defined at different levels in the hierarchy of a network. Depending on this definition, segment latency may include both link and element latency contributions. In this regard, [15] presents an exhaustive study of segmentation of latency between the RAN and CN. It considers multiple scenarios like network load, user distance, packet size, etc.

and presents the impact on latency. While the setup is designed to emulate real limitations on the network, these measurements are based on a lab setup. Nevertheless, this study is one of the most exhaustive segmentation efforts in the existing literature and most closely matches the problem statement addressed in this report.

While studies in the literature survey present the observed latency in different scenarios, 3GPP itself suggests certain latency numbers for uplink and downlink RAN communication in [16]. Downlink experiences a typical radio latency of 7.5 ms, while uplink experiences a typical radio latency of 17 or 12.5 ms (with or without a valid uplink grant). These numbers serve as the baseline for comparison with field measurements. While these are ideal values, the Living Edge Lab network used in our experiments should support similar latency values for the majority of the cases.

3 The Living Edge Lab Network

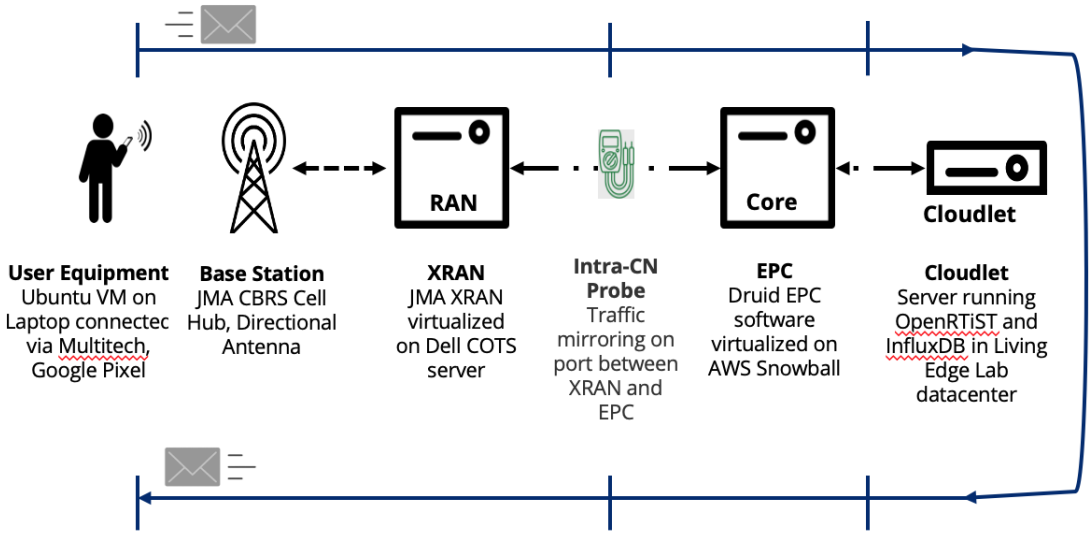

Figure 1: Living Edge Lab 4G LTE Network

The Living Edge Lab [17] deployed a private 4G LTE network designed to provide a low-latency setting to experiment with latency-sensitive edge computing applications. After becoming fully operational in June 2021, this network is used for research, education, and engagement with industry partners in edge computing.

The Living Edge Lab network operates on unlicensed GAA Citizens Broadband Radio Service (CBRS) spectrum at 3.5 GhZ. It is comprised of four base stations across Pittsburgh. Each base station consists of a JMA Wireless CBRS radio unit and directional antenna. The RAN backend processing unit is deployed using JMA Wireless software and is virtualized on a Dell COTS Server in the Living Edge Lab datacenter at the CMU campus. The Evolved Packet Core (EPC) is deployed using Druid EPC software and is virtualized using AWS Snowball Edge. Packets are

forwarded to a cloudlet [18] for computation, which is one-hop away from the EPC. We further instrumented the network with an Intra-CN probe, an internal server located in the Living Edge Lab data center that mirrors traffic on the ports between the XRAN and EPC and between the EPC and the cloudlet. The Living Edge Lab network is shown in Figure 1.

4 Latency Segmentation Framework

We set out to measure the latency introduced by each segment of the round-trip path of a packet in the Living Edge Lab network. We developed the latency segmentation framework to identify the sources of high latency in the network. Our goal is to use this knowledge to optimize the network and decrease the round-trip latency. The framework consists of three components: measurement, synchronization, and data management. To measure the latency of each segment, we instrumented the network with probes to collect traffic at the UE, XRAN, EPC and cloudlet. Table 1 details the specific segments of the network we measured. We were unable to measure the XRAN ingress to XRAN egress because of our inability to instrument the XRAN S1 link. We synchronized each probe to the same clock using NTP and PTP. The data management component involves configuring databases to store and retrieve information from observed packets at each probe and correlating each packet across every probe to use to calculate segment latencies. We developed a dashboard to monitor the network segment latencies in near real-time. The framework is shown in Figure 2 and further described in the sub-sections below.

Notation	Uplink Segment	Downlink Segment
UE-XRAN	UE egress to XRAN ingress	XRAN egress to UE ingress
XRAN-EPC	XRAN egress to EPC ingress	EPC egress to XRAN ingress
EPC-Cloudlet	EPC egress to Cloudlet ingress	Cloudlet egress to EPC ingress

Table 1: Segments Measured with Probes and Notation Used

4.1 User Equipment (UE)

We use both a Google Pixel 6 and a Windows laptop with an Ubuntu Virtual Machine as the UE for our latency segmentation experiments, alongside the network architecture described in Section 3. The Google Pixel is configured with a SIM card for the private network. We use a MultiTech MTCM2 MultiConnect microcell Cellular Modem adapter on the laptop to gain connectivity to the network. For benchmarking, we use ping and OpenRTiST [19], an application which generates a symmetric, data-intensive workload involving sending and receiving video frames from a server. The OpenRTiST server is running on the cloudlet in the Living Edge Lab data center.

4.2 Measurement

To determine the main segment contributors to round-trip latency in our system, we inserted four probes to collect packet details from traffic across the network. The probes, shown in Figure 2, are located at: the UE, Intra-CN probe, and the cloudlet. The Intra-CN probe contains two

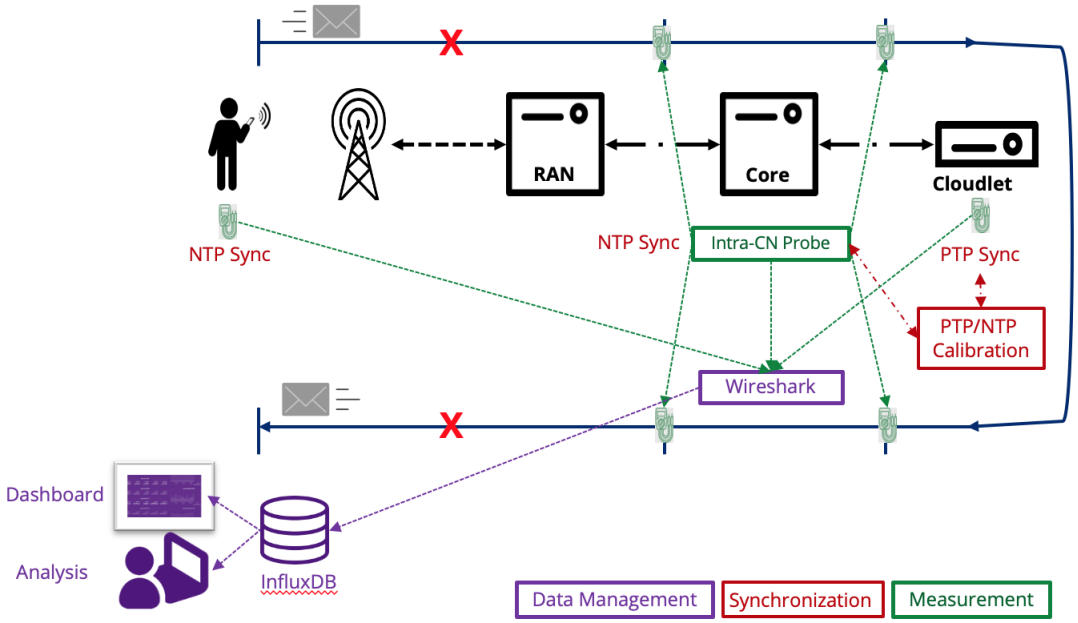

Figure 2: Instrumentation Architecture

measurement points, one to collect traffic between the XRAN and EPC and one to collect traffic between the EPC and the cloudlet. Each probe is responsible for executing a data collection script to collect incoming and outgoing traffic, later used to compute segment latency measurements.

4.3 Synchronization

To compute network segment latency values, each probe must synchronize to a common clock so the timestamps observed at each probe are consistent, and execute the data collection script to parse and store the observed packets. We use Network Time Protocol (NTP) and Precision Time Protocol (PTP) to synchronize the probes. The UE and Intra-CN probe are synchronized to the source using NTP, while the cloudlet is synchronized to the source using PTP. Both sources are machines located within the Living Edge Lab data center operating on the same clock. Although PTP offers more precision than NTP, we cannot use PTP at the Intra-CN probe or the UE due to the incompatibility of PTP with these probes. The UE does not support hardware timestamping, and the Intra-CN probe doesn't support hardware timestamping on the port compatible with PTP. Therefore, we would need to change the physical configuration of the network to support PTP at the Intra-CN probe and cannot support it at the UE.

Table 2 shows that the average offset for each probe is on the order of a few milliseconds. Additionally, the maximum average margins of error at the Intra-CN probe and cloudlet are 1ms or less (as reported by the `chrony sources` command[20]). In the context of round-trip latency in the tens of milliseconds, we viewed this offset and margin of error to be acceptable. The table shows that the UE probe has a higher offset and margin of error than the Intra-CN probe and cloudlet. However, in practice, we found the UE offset and margin of error to be significantly less

than the time coincident observed latency at the UE allowing us to consider this synchronization as also acceptable. Therefore, we decided to use this synchronization approach in our experiments.

Probe	Offset (μs)	Margin of Error (μs)
UE (Laptop)	-2265.286	10875.643
Intra-CN Probe	-11.580	1100.143
Cloudlet	-0.00264	0.0502

Table 2: Mean Probe Synchronization Offset of 20 Trials

4.4 Data Management

We extract and store packet-specific data to calculate segment latencies for the Living Edge Lab network. To calculate segment latencies, we record the time each packet arrives at every probe in the network. With each probe synchronized to the same clock, we use the difference in arrival timestamps for each packet at each probe to calculate segment latency. The process of calculating segment latencies involves: (1) storing identifying information and timestamps for each TCP and ICMP packet received at every probe, and (2) using the stored values to identify the same packet at each probe to calculate segment latencies for each packet. We predominantly use ICMP ping for benchmarking rather than OpenRTiST due to the complexity of correlating OpenRTiST frames on the UE and the overhead of running the OpenRTiST client on the laptop UE.

We first address the process of storing identifying information and timestamps for each packet. Using `tcpdump` [21] and `Pyshark` [22], we listen for traffic on the relevant interface for each probe, listed in Figure 3. We inspect each individual packet and extract the fields listed in Table 3. The extracted fields are stored in InfluxDB [23] databases, one table for TCP and one table for ICMP for each probe.

Field	Role
`src`	Source address for packet. Used to identify uplink vs. downlink packet.
`dst`	Destination address for packet. Used to identify uplink vs. downlink packet.
`time`	Time value. Used to uniquely identify packet.
`seqnum`	Sequence number. Used to uniquely identify packet.
`acknum`	(TCP-only) Acknowledgement number. Used to uniquely identify packet.
`epoch`	Time packet received, relative to Unix-epoch. Used for calculating segment latencies.

Table 3: ICMP and TCP Packet Fields

We conclude a packet is identical at each probe if the identifying fields, listed in Table 3, are identical for each probe's database entry. By performing a join on the identifying fields, we retrieve the timestamp for each individual packet at each probe. We then subtract the timestamps of adjacent probes to calculate the UE-XRAN, XRAN-EPC, and EPC-Cloudlet latencies, see Table 1 for more details on the measured segments. The resulting timestamps are stored in a separate database used for visualization. The data collection and correlation process is explained with a sample ICMP packet in Figure 3.

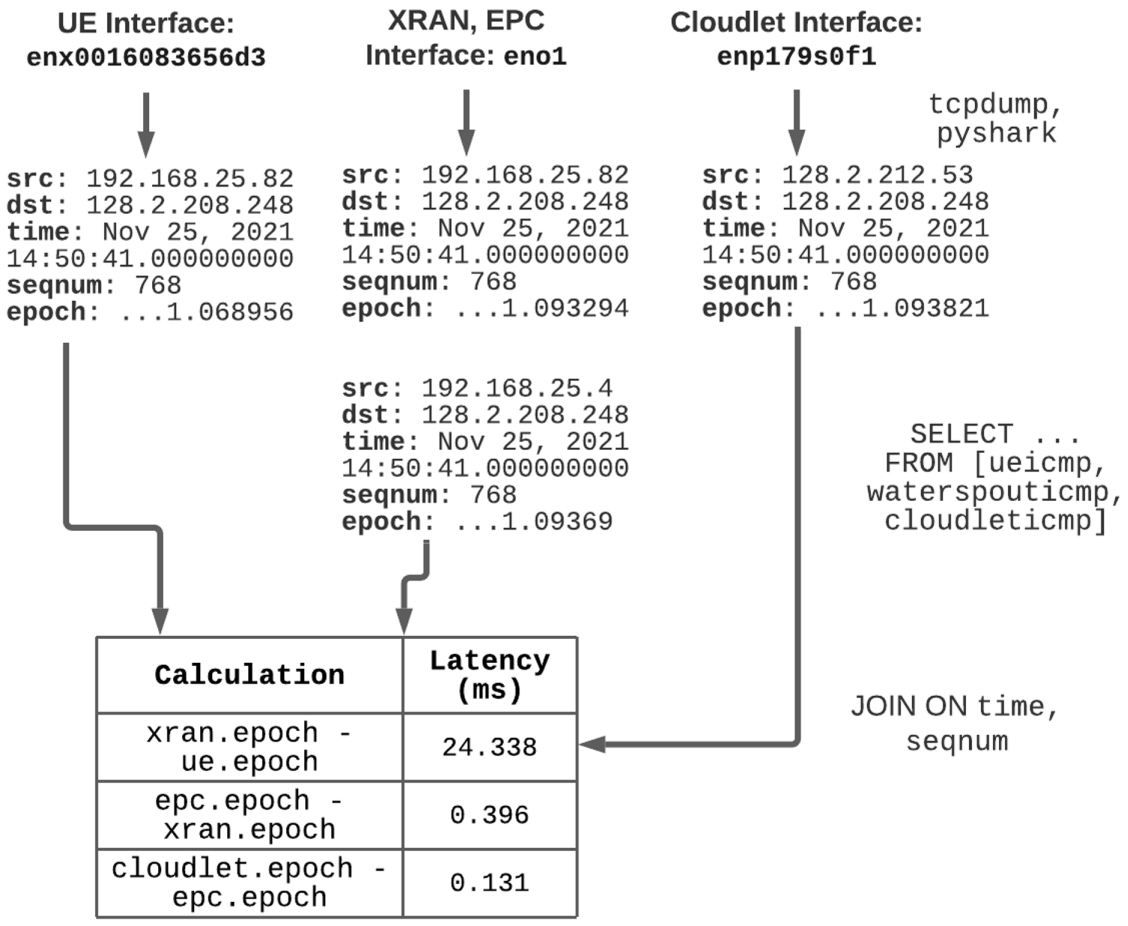

Figure 3: Data Packet Alignment Across the Network

4.5 Dashboard

To present the data in an accessible format, we visualize the results using Grafana [24]. The computed segment latency values are stored in an InfluxDB database. The Grafana dashboard displays near real-time summary statistics for the latency of each segment in the network. The calculated latency values are read from an InfluxDB table, with a delay of at most ten seconds between when a packet is observed in the network and when it appears on the dashboard. The dashboard can be used as an on-going network monitoring tool, where we see the near real-time latency of each segment in the Living Edge Lab network and determine where the sources of high latency are observed in the system. A screenshot of the Grafana dashboard from an ICMP trial is shown in Figure 4.

5 Experimental Results

This section describes the results of our experiments on the Living Edge Lab private network. We sought to understand how the network performed under a variety of different connectivity

Figure 4: Real Time Dashboard

scenarios and how it compared to commercial 4G LTE networks. We captured latency values for each segment across these scenarios to understand how latency is affected and to understand the sources of high latency in the Living Edge Lab network. We present the results of our experiments in the sub-sections below.

5.1 Latency Comparison of Commercial and Private Networks

We suspected the Living Edge Lab 4G LTE network would exhibit lower round-trip latency than the round-trip latency of a commercial network based on previous experiments [25] and the increased congestion on a commercial network. To confirm these results, we measured the round-trip latency of the Living Edge Lab network and the Mint Mobile commercial network. We collected 250 data points using ping with an interval of 100 ms at the Hillman Center Base Station for the Living Edge Lab network and on the commercial network. We used a Google Pixel 4 phone to measure the round-trip latency of the Living Edge Lab network and a OnePlus 6 phone connected to Mint Mobile to measure the round-trip latency of a commercial network. The results shown in Figure 5 confirm the round-trip latency of the Living Edge Lab 4G LTE network is approximately 50% less than the commercial network used in our experiments. This result is expected due to the additional traffic and scheduling overhead necessary to manage additional users in a commercial network. The Living Edge Lab network had less than five active users on the network at the time of our experiments. These results motivate the advantages of using a small-scale controlled LTE network for edge computing research.

5.2 Latency Segmentation Results

To determine the main segment latency contributor for the Living Edge Lab network, we measured the latency of each segment with the approach detailed in Section 4.

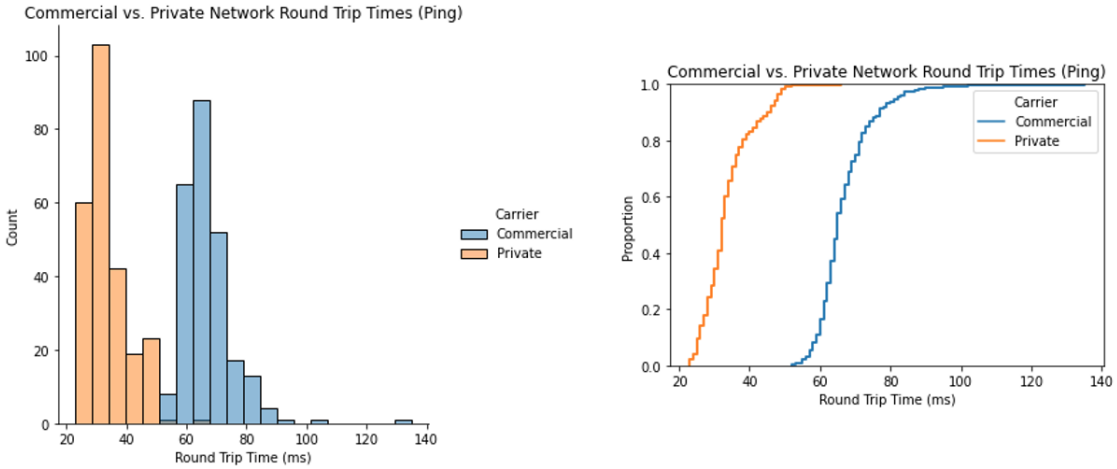

Figure 5: LEL vs. Commercial RTT Ping Times

We used ICMP ping running on the Windows laptop connected to the Living Edge Lab network as our test workload to generate network packet traffic. Ping packets were sent at an interval of 1000 ms, and we collected the data while stationary in a location of good signal coverage. We chose to use ICMP ping for these experiments because the low overhead of running ICMP ping minimizes the added delay from performing the experiments on a laptop rather than a mobile phone.

We previously tried to use OpenRTiST on the laptop, however, this approach was inadequate because the processing overhead from running the OpenRTiST Python client on the laptop adds an additional latency of tens to hundreds of milliseconds. We observed less overhead when using ICMP ping on the laptop. We also tried to use the Google Pixel for latency segmentation with ping or OpenRTiST but were unable to, due to our inability to instrument the phone to generate packet timestamps. Therefore, we use the laptop to collect packet timestamps at the device and calculate the segment latency of each section in the Living Edge Lab network.

With this latency segmentation approach, we confirmed the major source of latency in the round-trip latency is in the Device-XRAN (aka UE-RAN) segment of the network. The observed latencies for each segment are shown in Figure 6. In these histograms, we find that the observed latency for the Device-XRAN segment is significantly higher than the other measured segments. The other segments are minor contributors to the round-trip latency, which is typically observed to be tens of milliseconds in our experiments. Additionally, we observe that the uplink latency is noticeably larger than the downlink latency for the Device-XRAN segment, whereas the uplink and downlink spread is relatively similar for the XRAN-EPC and EPC-Cloudlet segments. Based on the 3GPP standards, the Device-XRAN segment is expected to incur overhead of approximately 17 ms uplink and 12.5 ms downlink [16], however, our observed uplink latencies are larger than the proposed standard values we expect to see. We suspect the additional latency is due to the difference in resource allocation methods for uplink and downlink. In LTE networks, uplink resource allocation is performed using single carrier frequency division multiplexing (SC-FDMA),

whereas downlink resource allocation is handled using orthogonal frequency division multiplexing (OFDMA). The uplink resource allocation consumes less power which is beneficial for mobile devices, however, it results in round-robin scheduling when waiting for the time slot to send the data since each timeslot allows only one user to send data. OFDMA is able to send data to multiple devices simultaneously. The variance we observed for the Device-XRAN uplink latency further corroborates our theory that scheduling mechanisms are leading to the large latency observed. The round-robin scheduling could cause a device to incur a significant amount of time waiting for its turn to send data which may lead to the spread in latency we found. We propose optimizations to decrease the latency of the major source of latency in Section 6. These optimizations have potential to both decrease the mean round-trip latency and decrease the jitter, which is particularly beneficial for quality of experience.

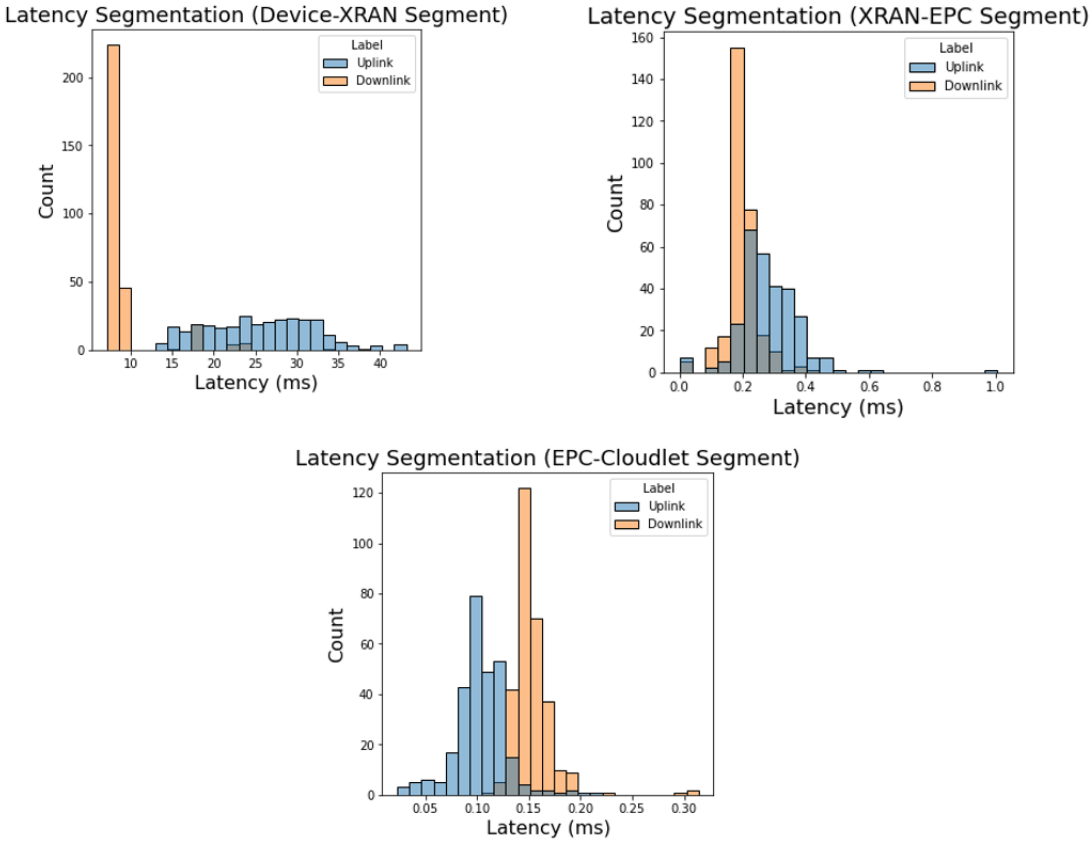

Figure 6: Uplink and Downlink Segment Latency Distributions

5.3 Further Experiments

We analyzed the effect of location, changing connectivity, handovers and increasing traffic on segment latency through additional field trials. Due to the signal variation when walking around, we were unable to reliably send packets from the laptop outdoors. Instead, we used the Google

Pixel for the outdoor trials. We also found correlating individual TCP packets with OpenRTiST frames to be too complicated, as each image frame is transmitted by multiple TCP packets. Therefore, for the field trials, we used ping with 1400-byte ICMP packets to emulate the size of TCP packets without incurring fragmentation. Since the mobile phone is unable to provide packet timestamps, as discussed in Section 4, to approximate Device-XRAN segment values we measured the round-trip latency, XRAN-EPC latency and EPC-Cloudlet latency. We were able to calculate an approximate value for Device-XRAN segment latency from the difference between the round-trip latency and the two recorded segment latencies. While this approach is not as precise as directly observing packets at the device and calculating segment latency as presented in Section 4, we find it provides a good approximation and sufficient information for our field trials due to the equipment limitations. The goal of our field trials was to perform a general assessment of how the network performs in different conditions, thus, precise values are not necessary for these trials. The results of the field trials are shown in the appendix.

6 Future Work

Our measurements showed that the primary contributor to round-trip latency is the UE-XRAN segment. The UE to XRAN uplink communication in particular takes the majority of time and can be optimized. There are various short term and long term measures for reduction of this latency. Firstly, the short term measures include tweaking the configurations in the XRAN to gain a better performance. A potential reason for more uplink communication latency is the skewed nature of allocated resources in the two directions. Historically, the UE receives more data on the downlink and sends less data on the uplink. This holds true for video streaming, internet browsing, etc. But for classes of applications like AR/VR, Autonomous Driving, etc. the UE could potentially transmit a live video feed on uplink and receive the processed response on downlink. Such applications will work better when the uplink and downlink are at least symmetric. One potential measure in this regard is to change the frame format running at the XRAN. By default, commercial LTE networks operate with a frame structure in TDD (Time Division Duplexing) which has more downlink slots than uplink. If a frame structure with equal opportunities in both directions is used, the communication will be balanced and latency can potentially reduce. However, this latency difference will never reduce to zero because of processing overhead in the uplink. Everytime a UE tries to transmit data, it has to first request access to wireless resources and, thus, sends a scheduling request. Upon receiving the grant from XRAN, it sends a status report of the data it has and the XRAN accordingly schedules the UE. In contrast, this first step to requesting a grant is not present in downlink communication.

Nevertheless, there is a technique to reduce the overhead incurred in sending a Scheduling Request (SR). Semi-Persistent Scheduling (SPS) [16] can be used to gain the scheduling grant for a prolonged duration. This way, the UE can skip the initial step and can get scheduling opportunities just by sharing its buffer status.

Mobility can also incur occasional spikes in latency due to handover between multiple eNodeB. A technique called Random Access Channel (RACH)-less handover [16] can reduce the time required in switching the connectivity of UE from one eNodeB to the other.

Long term measures include a move to latest technologies like 5G-NR (New Radio). It has a RAN specially developed with latency considerations. The radio units expected for 5G have more processing capacity and expected to achieve round-trip communication within 10 ms.

We also observed significant uplink latency variation (jitter) between the UE and the XRAN. Excessive jitter can have a strong negative impact on some edge-native applications. Understanding the causes of this jitter and potential mitigation approaches remains a future goal. It may be that the scheduling approach discussed above also contributed to the jitter.

Our segmentation measurements were constrained by where we could practically introduce measurement probes. In particular, further segmenting the UE-XRAN segment to get greater visibility into the sources of latency within that segment remains a future goal. We are investigating methods for placing probes within the XRAN to directly observe traffic within the S1 GTP tunnel.

7 Conclusion

This report presents our efforts in segmenting the Living Edge Lab 4G LTE network with measurement probes to determine the main contributor to segment latency in communication. With probes at the UE, XRAN, EPC and cloudlet actively synchronized to the same clock, we performed a detailed study of field trials and observations of network traffic. We found the uplink communication on the UE-XRAN segment to be the primary delay element in the round-trip latency, in agreement with the expected values from the 3GPP standards [16].

Our network instrumentation remains in place at the Living Edge Lab data center and can be used with our Grafana dashboard to monitor the network status in near real-time. We propose multiple optimizations to encourage future research in decreasing the Living Edge Lab round-trip latency. These optimizations are intended to promote more efficient edge computing applications and support for latency sensitive applications.

The latency segmentation framework code is available at [26].

8 Acknowledgements

This work was conducted at the Carnegie Mellon University Living Edge Lab in Pittsburgh, Pennsylvania USA under the leadership of Professor Mahadev Satyanarayanan. The authors would like to thank Professor Satyanarayanan, Professor Siewiorek and Professor Smailagic for their guidance on this project during 15-821/18-843, the Mobile and Pervasive Computing Course where this project was completed.

We would like to thank Crown Castle, Amazon Web Services, and JMA Wireless for their extensive support and expertise in the design and deployment of the Living Edge Lab Network.

This research was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0051 and by the National Science Foundation (NSF) under grant number CNS-1518865 and the NSF Graduate Research Fellowship under grant numbers DGE1252522 and DGE1745016. Additional support was provided by Intel, Vodafone, Deutsche Telekom, Crown Castle, InterDigital, Seagate, Microsoft, VMware and the Conklin Kistler family fund. Any opinions, findings, conclusions or recommendations expressed in this material are those

of the authors and do not necessarily reflect the view(s) of their employers or the above funding sources.

9 Appendix

9.1 Connectivity Trials

For the connectivity field trials, we analyzed the effect of constant, improving, and worsening connectivity on segment latency. The constant connectivity trial involved standing in place near each of the four base stations to achieve a baseline trial of ideal connectivity. We then emulated improving connectivity by walking from a location of poor connectivity towards the base station. The worsening connectivity trial was conducted by walking away from the base station to a location of poor connectivity. The results are shown in Figures 7, 8, 9. Each latency measurement shown below consists of the sum of uplink and downlink latencies to represent the round-trip latency.

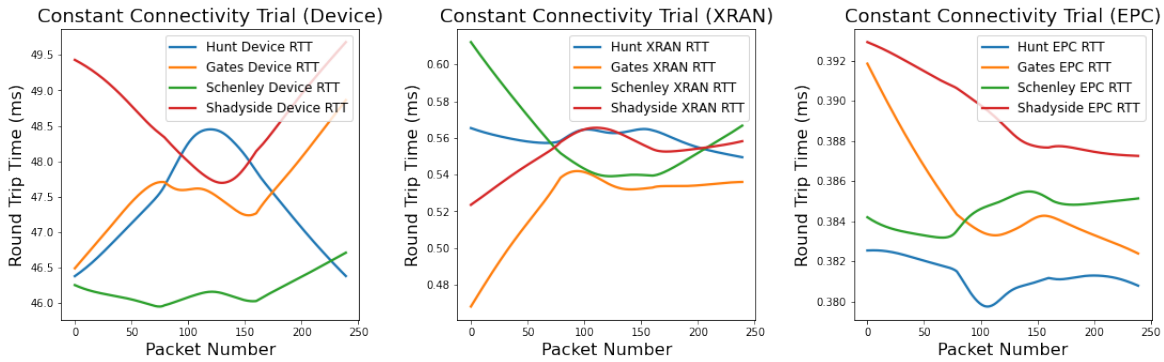

Figure 7: Segment Latencies Observed for Constant Connectivity Trial

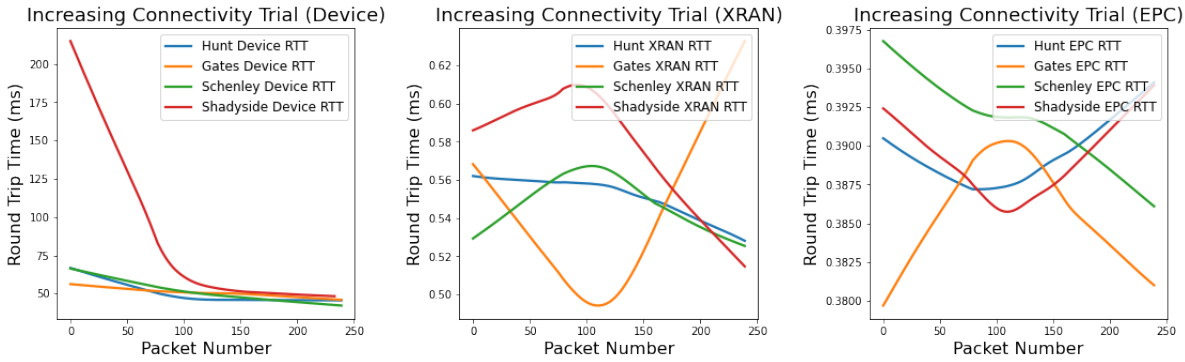

Figure 8: Segment Latencies Observed for Improving Connectivity Trial

The results observed agree with our expectations, as we find significant increases in latency as we reach a location of worse connectivity and significant decreases in latency as we improve connectivity. There are some differences in latency observed at each base station, however, the trends are consistent across each location. We suspect the minor fluctuation of latency in the

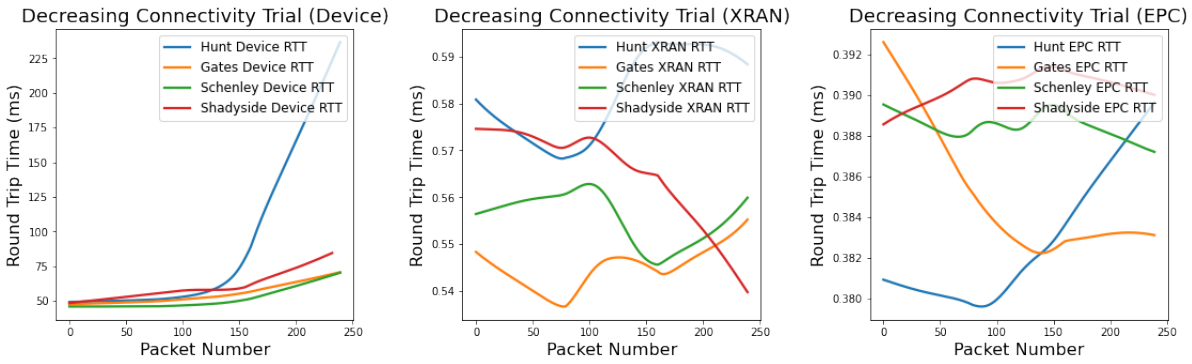

Figure 9: Segment Latencies Observed for Worsening Connectivity Trial

constant connectivity experiment is due to variability in the network and environment during the experiment.

9.2 Handover Trial

We tested the effect of a handover from the Hunt Base Station to the Hillman Center Base Station. This handover was achieved by gradually walking from the former base station to the latter base station with the mobile phone. The results are shown in Figure 10, where each segment latency includes the sum of uplink and downlink latency.

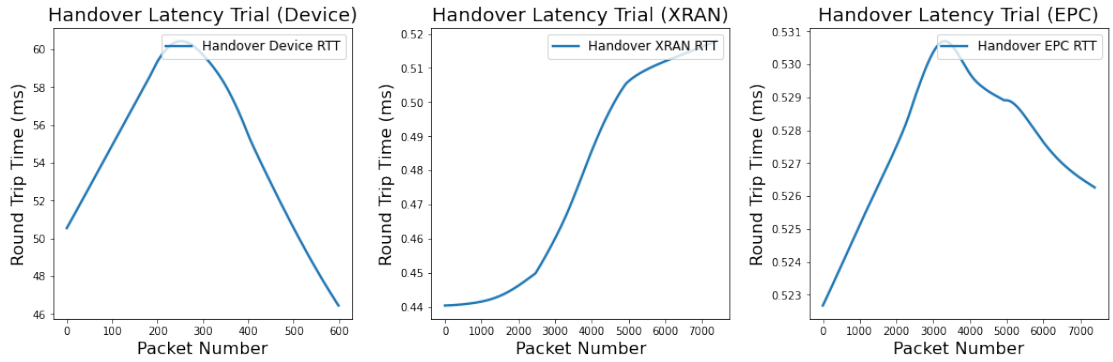

Figure 10: Segment Latencies Observed for Handover Trial

We noticed a significant spike in latency as we walk from the Hunt Base Station to the Gates and Hillman Center Base Station. We suspect some of this latency increase is due to the decreased connectivity both by walking through an outdoor corridor away from the Hunt Base Station and not having line-of-sight access to the Hillman Base Station, however, we suspect some increase in round-trip latency is due to the handover. A handover is known to add 15.6 ms to the round-trip latency [13]. We suspect the significant spike in latency is both from the decreasing connectivity by traveling away from the Hunt Base Station and towards the Hillman Center Base Station and the handover overhead. However, with our experiments, we have no way to be certain whether the effect of a handover is observed. We believe the increase and subsequent decrease in round-trip

latency as we travel away from the Hunt base station confirms we have switched from the Hunt Base Station to the Hillman Center Base Station.

9.3 Traffic Trial

We tested the effect of increasing traffic at the laptop by sending 200 ICMP ping packets from the laptop to the cloudlet. After 100 ICMP ping packets were sent, we started OpenRTiST on the laptop to produce competing TCP traffic on the network. The resulting ICMP round-trip latency for each segment is shown in Figure 11.

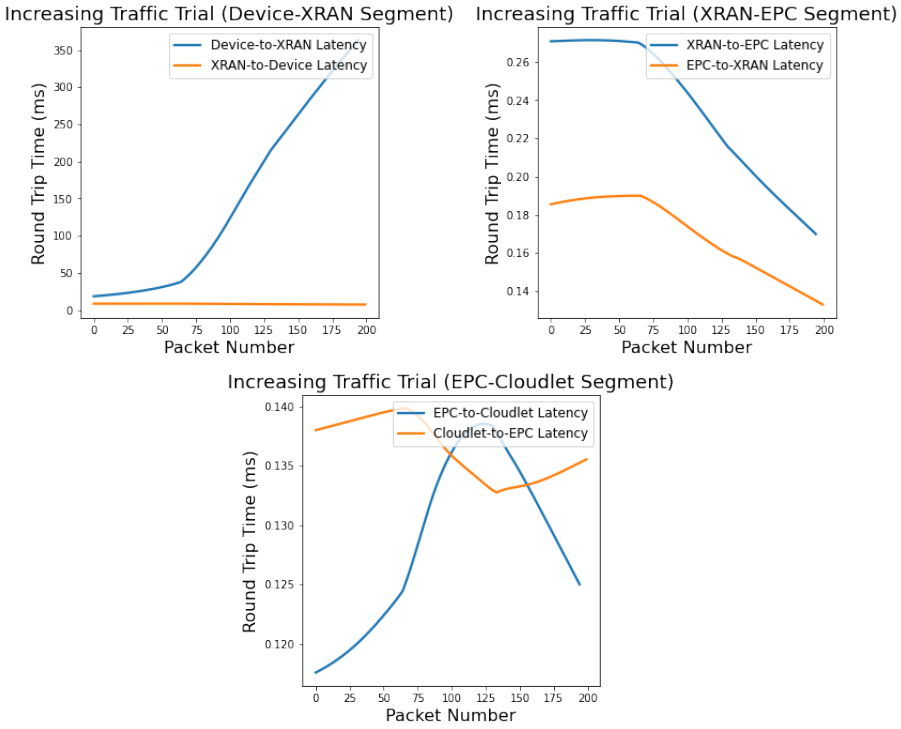

Figure 11: Segment Latencies Observed for Increasing Traffic Trial

After OpenRTiST started, we noticed a significant increase in round-trip latency at the laptop-XRAN segment. We suspect this may be due to packet queueing at the device, since the laptop is resource-constrained, and queueing at the XRAN, since the number of uplink slots is fewer than the number of downlink slots at the eNodeB, yet the traffic for these experiments is symmetric with uplink data quantity and downlink data quantity sent. Additionally, the significant increase in uplink latency contrary to downlink latency suggests that we have disproportionate numbers of uplink and downlink slots allocated to the device at this moment. This suggests we may not have a sufficient number of uplink slots to send both ping and OpenRTiST traffic without incurring significant latency overhead.

References

[1] TR 36.401. *LTE; Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Architecture description*. Standard. 3GPP, 2020.

[2] *Wireshark*. URL: https://www.wireshark.org.

[3] Shuohua Guo James Lin and Yang Zhang. "The Performance Analysis of LTE Network, ENSC 427: Communication Networks Spring 2014". Simon Fraser University, 2014. URL: https://www.sfu.ca/~ljilja/ENSC427/Spring14/Projects/team6/ENSC427_team6_report.pdf.

[4] Marilynn P Wylie-Green and Tommy Svensson. "Throughput, capacity, handover and latency performance in a 3GPP LTE FDD field trial". In: *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*. IEEE. 2010, pp. 1–6.

[5] Hilson G. Vilar De Andrade, Caio C.L.L. Ferreira, and Abel G. Da Silva Filho. "Latency Analysis in Real LTE Networks for Vehicular Applications". In: *2016 VI Brazilian Symposium on Computing Systems Engineering (SBESC)*. 2016, pp. 156–161.

[6] Yuzhe Xu. "Latency and Bandwidth Analysis of LTE for a Smart Grid". MA thesis. KTH Royal Institute of Technology, 2011. URL: https://www.diva-portal.org/smash/get/diva2:565509/FULLTEXT01.pdf.

[7] Niwas Maskey, Seppo Horsmanheimo, and Lotta Tuomimäki. "Latency analysis of LTE network for M2M applications". In: *2015 13th International Conference on Telecommunications (ConTEL)*. 2015, pp. 1–7.

[8] Shilpa George et al. "OpenRTiST: end-to-end benchmarking for edge computing". In: *IEEE Pervasive Computing* 19.4 (2020), pp. 10–18.

[9] Bhaskar Prasad Rimal, Martin Maier, and Mahadev Satyanarayanan. "Experimental testbed for edge computing in fiber-wireless broadband access networks". In: *IEEE Communications Magazine* 56.8 (2018), pp. 160–167.

[10] Zhuo Chen et al. "An empirical study of latency in an emerging class of edge computing applications for wearable cognitive assistance". In: *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*. 2017, pp. 1–14.

[11] Wenlu Hu et al. "Quantifying the impact of edge computing on mobile applications". In: *Proceedings of the 7th ACM SIGOPS Asia-Pacific Workshop on Systems*. 2016, pp. 1–8.

[12] Junjue Wang et al. "Towards scalable edge-native applications". In: *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*. 2019, pp. 152–165.

[13] Deepti Singhal et al. "LTE-Advanced: Handover interruption time analysis for IMT-A Evaluation". In: *2011 International Conference on Signal Processing, Communication, Computing and Networking Technologies*. 2011, pp. 81–85.

[14] Nabil Mesbahi and Hamza Dahmouni. "Delay and jitter analysis in LTE networks". In: *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*. 2016, pp. 122–126.

[15] Ashish Kurian. "Latency analysis and reduction in a 4G network". MA thesis. Delft University of Technology, 2014. URL: https://repository.tudelft.nl/islandora/object/uuid%3Ae1badd8d-a384-49a1-b958-a0c1e499c539.

[16] TR 36.881. *Study on latency reduction techniques for LTE*. Standard. 3GPP, 2016.

[17] Carnegie Mellon University. *Edge Computing @ CMU Living Edge Lab*. 2022. URL: https://www.cmu.edu/scs/edgecomputing/index.html.

[18] Mahadev Satyanarayanan et al. "The Case for VM-Based Cloudlets in Mobile Computing". In: *IEEE Pervasive Computing* 8.4 (2009), pp. 14–23. DOI: 10.1109/MPRV.2009.82.

[19] cmusatyalab. *OpenRTiST: Real-Time Style Transfer*. URL: https://github.com/cmusatyalab/openrtist.

[20] Oleg Obleukhov. *Building a more accurate time service at Facebook scale*. 2020. URL: https://engineering.fb.com/2020/03/18/production-engineering/ntp-service/.

[21] The Tcpdump Group. *TCPDUMP and LIBPCAP*. 2022. URL: https://www.tcpdump.org/.

[22] KimiNewt. *pyshark*. URL: https://github.com/KimiNewt/pyshark.

[23] InfluxData Inc. *influxdata*. 2022. URL: https://www.influxdata.com/.

[24] Grafana Labs. *Grafana*. URL: https://grafana.com.

[25] James Blakley. *The Quest for Lower Latency: Announcing the New Living Edge Lab Wireless Network*. 2021. URL: https://www.openedgecomputing.org/the-quest-for-lower-latency-announcing-the-new-living-edge-lab-wireless-network/.

[26] The Living Edge Lab. *LEL Latency Segmentation Framework*. 2022. URL: https://github.com/cmusatyalab/NetworkLatencySegmentation.