Open Monitoring Platform for Mobile Broadband

Wolfgang Hofer*, Philipp Svoboda*, Wolfgang Kastner†, Vaclav Raida* and Markus Rupp*
*Institute of Telecommunications, TU Wien, Austria
†Institute of Computer Engineering, TU Wien, Austria
wolfgang.e389.hofer@tuwien.ac.at, {firstname.lastname}@tuwien.ac.at

***Abstract*—Enhanced Mobile Broadband (eMBB) is just around the corner, and its performance aspect makes it interesting for Vehicle-to-Vehicle (V2V) communication. Fair benchmarking of communication networks has been a research topic for decades. In contrast to wired networks, mobile networks lack a quick, accurate, and data-saving tool to describe the properties of the connection in a vehicle. Therefore, this work aims to create a crowdsourcing-ready and easily expandable software platform capable of analyzing the current network conditions in a quick and data-saving way. Based on literature research, we identified *CRUSP* as quick, data-saving, and accurate method to characterize mobile communication networks. Afterwards, we realized *Constant Rate Ultra Short Probing* (CRUSP) in a flexible software platform for performing, gathering, and analyzing measurements. The evaluation of the software was performed in a configurable and shielded reference cell and delivered an average error of 2.44% to iPerf3, a popular reference tool. In conclusion, the results indicated that the developed software operates accurately in Long-Term Evolution (LTE) networks.**

***Index Terms*—available bandwidth, ABW, eMBB, mobile broadband, measurement platform, CRUSP**

I. Introduction

In the European Union, the digital transformation is a major goal, reflected in demand for omnipresent access to mobile services via a digital link of high quality of service. Today, nomadic Internet users rely on fast, reliable, and dependable internet connections anywhere at any time. However, in mobile network deployments, factors such as cell density, pathloss, and interference are, among others, limiting factors in terms of performance.

Especially in rural areas, mobile networks offering services used in vehicular scenarios face two challenges, sparse deployment, and large pathloss. However, these areas are of special interest in the upcoming technologies, as Vehicle-to-Vehicle (V2V) communication on high layer roads is one major use case in the current discussion.

Applications on vehicles come with distinct Quality of Service (QoS) requirements. For example, car drivers often struggle to overtake large vehicles. They can get stuck behind a truck, being unable to see in front of it. In order to solve this problem, Gomes et al. [1] suggested leveraging V2V communication for a see-through-system. Thereby, the truck operates a front camera and streams its footage to the vehicle behind it. The see-through-system then projects a 3D model of the road and landscape extracted by the footage onto the windshield of the car as if the truck is transparent. In order to match projection and reality, the latency must be kept very low. Furthermore, the V2V connection must provide enough digital bandwidth to transmit the data in sufficient quality. Fortunately, the advent of the fifth generation of mobile communication networks (5G) with enhanced Mobile Broadband (eMBB) creates the technical basis for this kind of V2V communication.

However, in mobile networks, resources are shared among all users and offered dynamically [2]–[5]. Consequently, the available bandwidth is not constant but a function of time and space.

Indeed, current tools to assess the available bandwidth are often slow, data-hungry, or inaccurate when deployed in mobile networks. Consequently, there is a need for a platform that provides an agent-based and fair benchmarking of mobile broadband networks.

This work aims to provide an open-source ready-to-use benchmarking platform, where researchers can plug in an implementation of a custom network characterization method. Furthermore, the platform aims to store, access, filter, sort, and download the performed measurements. As a result, this platform is a ready-to-use tool for future research on network characterization in V2V communication.

In order to fulfill this vision, we selected resourceful bandwidth measurements as our use case. After choosing *Constant Rate Ultra Short Probing* (CRUSP) [6] as an appropriate method for characterization, we implemented it in the benchmarking platform. The result is an Android application and a scalable system capable of estimating the available bandwidth in a split second. All measurements are stored in a database for analysis. Additionally, developers can extend the platform with custom network benchmarking implementations through well-defined interfaces.

The paper is structured as follows. At first, Section II analyzes the state of the art in benchmarking mobile communication networks. Section III deals with the setup and architecture of the software platform. Section IV presents the results of the validation process. Finally, Section V summarizes the findings of this thesis. Furthermore, it takes a look at limitations and future work.

II. State of the Art

A well functioning network is a result of constant optimization at all different layers of the network. Monitoring the current state of the networks is an important enabler for this optimization process. This monitoring requires a fast, resourceful way of probing the network and the user's location,

978-1-7281-5207-3/20/$31.00 ©2020 IEEE

e.g., a framework that allows for crowdsourcing performance measurements on mobile end-user devices.

There already exist projects that want to offer an easily accessible and flexible monitoring platform. For example, by regulation bodies like RTR[1] or by the MONROE-project[2]. However, they come with a few shortcomings. Either they lack the flexibility to implement a custom measurement method, or they lack a ready-to-go version with a quick setup. Furthermore, those platforms often lack a volume limit for repeated measurements. If the data traffic exceeds the data volume limit, the repeated measurements should stop. Therefore we take a step back and start with a closer look at the characterization of mobile broadband networks.

A. Characterization in (e)MBB

In order to determine the QoS of a connection, four characteristics are vital [7, p. 405]: *bandwidth*, *latency*, *jitter*, and *packet loss*. However, not all of them are equally useful to the user. Indeed, users are hardly interested in jitter and packet loss. Moreover, quick and useful tools for latency measurement already exist (e.g., ping). However, considering the mobile broad band usage scenario, the essential characteristic is the digital bandwidth.

Unfortunately, applications still need a non-intrusive and quick way to actively measure the digital bandwidth. Non-intrusive means, the application injects no additional traffic or a small amount into the network. For example, traditional estimation tools like Ookla's Speedtest [8] can consume around 100 MB and more. In contrast, non-intrusive techniques inject only a small fraction of it (less than 1 MB).

Various metrics are associated with the term *bandwidth*. Available bandwidth (ABW) is one of them and "the ideal choice to characterize the network path" [9]. It is the unused bandwidth available on the tight link after subtracting the cross-traffic from the capacity [10].

However, current tools to assess the bandwidth are often slow, intrusive, or inaccurate when deployed in mobile networks. Furthermore, they often require system access in order to execute them on the command line. The next subsection gives an overview of currently available tools to measure the ABW.

B. Available Measurement Tools

The Google Play Store and Apple App Store list dozens of apps to measure the ABW. The most downloaded Android application (more than 100.000.000) is *Speedtest* by Ookla [8]. It measures, among other characteristics, the ABW and latency. However, its downlink and uplink measurements consume dozens of MB and take at least 10 seconds each.

[1]https://www.rtr.at/en
[2]https://github.com/MONROE-PROJECT

Name of App	ABW	Latency	Downlink Test Duration	Open-Source
Speedtest (by Ookla)[3]	✓	✓	Long	
RTR NetTest (oRMBT)[4]	✓	✓	Avg.	✓
Speedcheck (by Etrality)[5]	✓	✓	Avg.	
Meteor (by OpenSignal)[6]	✓	✓	Long	
NDT Test (by M-Lab)[7]	✓	✓	Long	✓
fast.com (by Netflix)[8]	✓		Long	
iPerf3[9]	✓	✓	Long	✓

TABLE I: State of the art applications

As Table I shows, this trend manifests itself in all popular tools available for download. They take at least five seconds where "Avg." represents a measurement duration between 5.00 sec - 9.99 sec, and "Long" indicates a duration equal or longer than 10 sec. Furthermore, those tools inject a considerable amount of data into the network during each measurement. Moreover, developers will have a hard time to extend the mentioned tools with their custom measurement method.

In summary, the long duration and data-hungry behavior makes these tools unsuitable for quick and non-intrusive measurements. Therefore, researchers already started decades ago with the development of lightweight methods to estimate the ABW.

C. Estimation Methods

Methods estimating the ABW should be *accurate*, *non-intrusive*, quick, and representative for the service quality. Table II presents an excerpt of methods to estimate the ABW. Numerous of them have been developed with focus on wired networks or WiFi. However, mobile networks differ from wired networks or WiFi in several aspects. For example, in mobile networks, IP packets often arrive in bursts [11].

Name of the Method	Quick	Non Intrusive	Open-Source Application	Mobile
pathchirp [12]	✓	✓	✓	
Wbest [13]	✓	✓	✓	
NextFit [14]		✓		✓
PathQuick3 [15]	✓	✓		✓
CRUSP [6]	✓	✓	✓	✓

TABLE II: Methods to estimate the ABW

[3]http://www.speedtest.net
[4]https://www.netztest.at/en/Test
[5]https://www.speedcheck.org
[6]https://meteor.opensignal.com
[7]https://www.measurementlab.net/tests/ndt
[8]https://fast.com/
[9]https://github.com/esnet/iperf

Traditional methods such as pathchirp [12] or Wbest [13] are not sufficiently accurate in mobile networks. Unfortunately, methods developed explicitly for mobile networks such as NextFit [14] and PathQuick3 [15] also produce errors of around ten percent. The best results, while still being quick and non-intrusive, delivers CRUSP. Therefore, we picked CRUSP for implementation.

In short, CRUSP sends UDP datagrams at a constant rate from a server to a client. Due to the nature of mobile networks, these datagrams arrive in bursts. CRUSP detects these bursts and calculates the ABW for each one by dividing its accumulated data volume by its elapsed time. In the end, CRUSP builds the average ABW over all bursts. Raida et al. [6] provides a detailed description of the calculation in their work.

However, the platform to be developed should also offer easy expandability for other methods. The next section will present the architecture providing a flexible platform.

III. Open Monitoring Platform for Mobile Broadband

At the beginning of a software project, the developers collect all relevant requirements from the stakeholders. These requirements provide a solid overview of the capabilities of the software product.

A. Requirements

A suitable tool to describe functional requirements are user stories. In short, a user story focuses on why and how the user interacts with the software [16]. The following ten summarize the scope of the project.

A user wants to...

1) start a measurement on a user equipment (UE).
2) see previous measurement results on a UE.
3) measure on a UE continuously in a predefined interval.
4) see the measurement in the foreground of the UE.
5) set a data volume limit for continuous measurements, and if the limit is exceeded, the measurements stop.

An engineer wants to...

6) change settings on a UE.
7) change and provide default settings in the backend to create customized experiments.

An analyst wants to...

8) store each measurement permanently on the server.
9) filter and sort the results by its properties.
10) export selected measurements to process them further.

Functional requirements express the behavior of the software product. In contrast, non-functional (or non-behavioral) requirements define software quality attributes. The following list presents an overview of the gathered non-functional requirements.

1) **Scalability**: The architecture should be scalable to serve a growing number of users.
2) **Performance**: No user should wait longer than one second for a measurement request.
3) **Performance**: No analyst should wait longer than one second for a filtering or sorting request to the database.
4) **Performance**: A user should be able to perform measurements with data rates up to 1 GB/s.
5) **Open-Source**: The software project should be free to access.
6) **Privacy**: The software project needs to be compatible with the General Data Protection Regulation.
7) **Expandability**: Well-defined interfaces should provide easy expandability for further measurement methods.

B. Technologies and Architecture

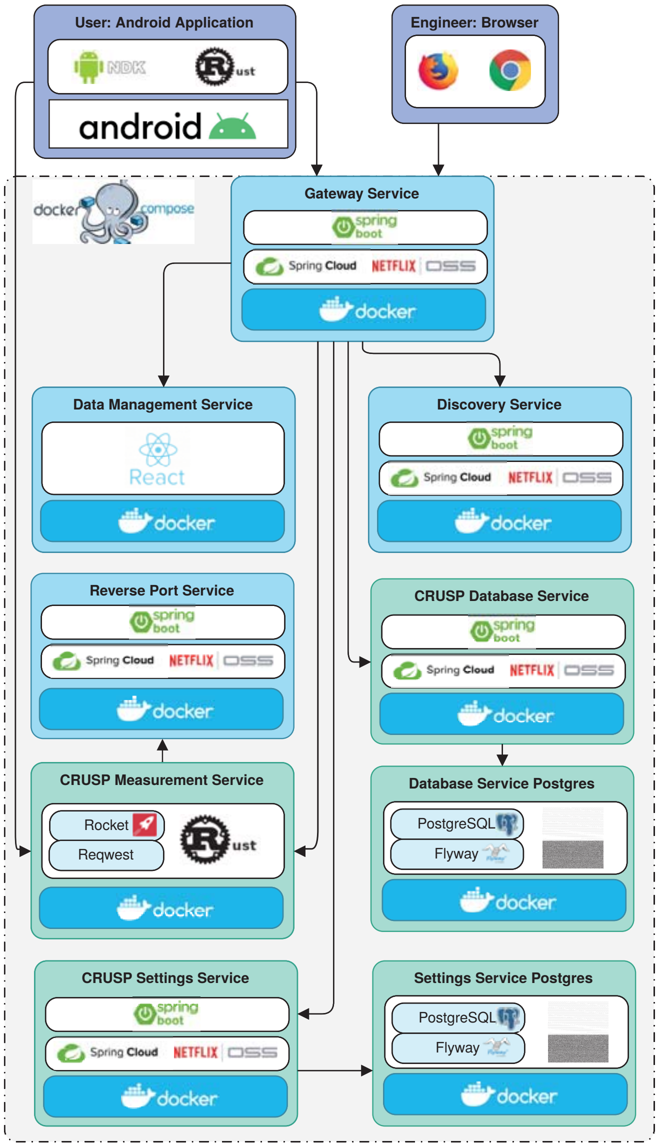

Fig. 1: Architecture and technology stack

The software framework builds on a microservices architecture in the backend for flexibility, scalability, and maintainability. Thereby, the microservices architectural style is “an approach to developing a single application as a suite of small services, each running in its separate process and communicating with lightweight mechanisms, often an HTTP resource API" [17]. Figure 1 gives an overview of the architecture and technology stack. In total, we defined nine services running in the backend in Docker[10] containers.

The blue services build the basic scalable structure of the system. The API gateway (Gateway Service) forwards all incoming traffic to the responsible service. The Discovery Service holds an overview of all services in the system. Those services use Java with the Spring Boot framework[11]. Additionally, Spring Cloud[12] components and the Netflix Open-Source Software Center (OSS) enable scalability features of the software.

In contrast, the green services represent the customizable part of the framework with CRUSP as example measurement method. Thereby, the system divides the measurement aspect into two parts. First, the measurement settings with the CRUSP Settings service (incl. Postgres[13]). Secondly, the measurement part with the CRUSP Measurement Service and Database Service (incl. Postgres). Unlike the other services, the Measurement Service is implemented in the programming language Rust[14]. This language combines memory safety with similar speed as C++ in order to overcome Java's accuracy problems when time-stamping in the nanoseconds range.

Finally, the frontend of the platform in purple consists of a measurement application and an analysis tool. The measurement application runs on Android and integrates a Rust client via the Native Development Kit (NDK). In contrast, the analysis tool allows access, filtering, sorting, and download of performed measurements. It uses Facebook's React[15] framework for easy access to the results via a browser.

The open-source project is hosted at gitlab.com[16]. In order to add a new measurement method, simply implement the method and run it in a docker container. After registering at the Discovery Service and adding the routes to the Gateway Service, the measurement service is ready to go.

IV. Evaluation

In order to show the accuracy of the developed platform, we conducted a series of measurements using the CRUSP module with the platform. For reproducibility, they took place in a shielded reference cell with a constant cell load. In there, we could adjust the attenuation level freely.

A. Test Setup

In short, the setup consisted of a server in the university network, a reference cell, and an anechoic chamber. Figure 2 depicts all vital components of the setup.

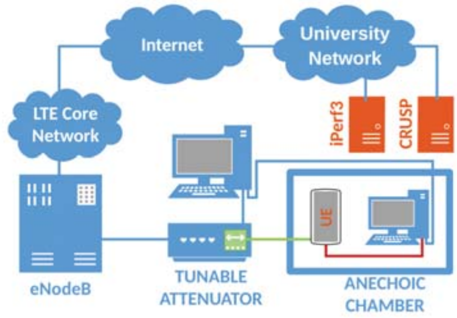

Fig. 2: Network setup

In the experiment, we examined the accuracy of the platform in an LTE network. Therefore, a CRUSP configuration with a data volume of $0.93MB$ and a packet size of $1kB$ per estimation was deployed. The calculation for the data rate divides two datagrams into different bursts if their timestamp has a time gap $\Delta_{\min}$ larger of 500 ms. Furthermore, the platform logged the RSRP-value for each measurement in order to compare the results based on their signal strength. Thereby, RSRP stands for *Reference Signal Received Power* and represents the strength of the LTE signal. Additionally, we took reference measurements with the well-known tool *iPerf3* [18].

[10]https://github.com/docker
[11]https://github.com/spring-projects/spring-boot
[12]https://github.com/spring-cloud
[13]https://github.com/postgres/postgres
[14]https://github.com/rust-lang/rust
[15]https://github.com/facebook/react
[16]https://gitlab.com/gitlabwolf/infrastructure-for-mobile

B. Test Procedure

This experiment went through the following steps:

1) SET the CRUSP configuration in the Android App
2) SET the attenuation level at the base station to $52dB$
3) WHILE the attenuation level is $\leq 78dB$
   a) PERFORM one iPerf3 measurement for 10 seconds
   b) PERFORM ten CRUSP measurements with the Android App
   c) INCREASE the attenuation level by $2dB$

C. Results

As Figure 3 (a) visualizes, the estimations of the platform with the CRUSP module are similar to the iPerf3 reference measurements. Figure 3 (b) shows a mean error of 2.44% to the iPerf3 results. Moreover, the error appears to be independent of the RSRP. However, the measurement platform seems to overestimate at low RSRP-values (RSRP $< -108dBm$) slightly.

The platforms CRUSP estimations achieve similar results compared to the 10-second iPerf3 measurements in an LTE network. This result coincides with the experiments on CRUSP by Raida et al. [19].

D. Limitations and Future Work

Despite finding a suitable method to determine the ABW, this work has some limitations. Indeed, we verified the accuracy in an LTE network. However, the platform lacks the evaluation in a 5G network. Future work could take the developed platform and test it for its accuracy in UMTS and

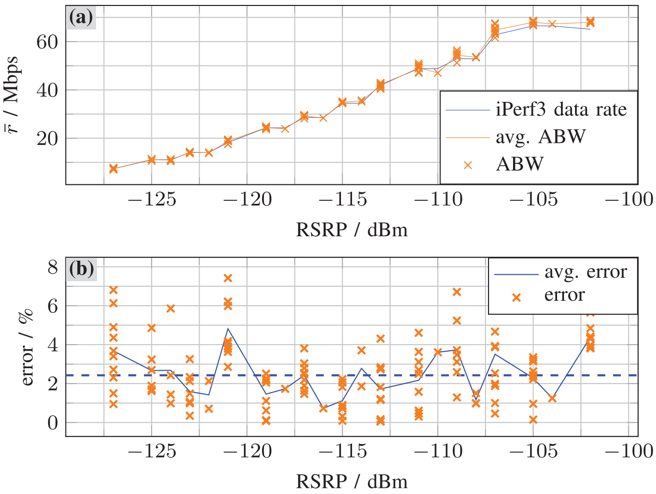

Fig. 3: (a) CRUSP module vs iPerf3 for each RSRP-value (b) error of CRUSP estimations to iPerf3 reference measurement for each RSRP-value

eMBB networks. Further research is needed for validation under different network conditions.

Additionally, this system can act as a data-gathering tool. Therefore proper load tests are needed to verify the scalability of the platform.

V. Conclusion

At first, we analyzed the state of the art in the characterization of mobile communication networks and identified CRUSP as quick, non-intrusive, and accurate method for ABW estimation. Next, we developed a platform for fair benchmarking of (e)MBB mobile communication networks based on CRUSP. In the backend, we created a scalable system that uses a microservices architecture that is flexible enough to develop and integrate new measurement methods easily. This system can persist, filter, sort, and export the performed measurements via an analysis tool. The Android application creates the measurements in a fraction of a second by pressing a button.

The evaluation in terms of accuracy took place in a configurable and shielded reference cell. Thereby, repeated measurements of the ABW led to an average error of 2.44% to the well-known reference tool iPerf3 in a standard LTE cell. Compared to traditional tools, this is an excellent result considering that only a fraction of the time (around 150 ms) and data (930 kB) was used.

Acknowledgment

This work has been funded by the ITC, TU Wien. The financial support of the Austrian BMWFW and the National Foundation for Research, Technology, and Development is gratefully acknowledged. The research has been cofinanced by the Czech GA CR (Project No. 17-18675S and No. 13-38735S) and by the Czech Ministry of Education in the frame of the National Sustainability Program under grant LO1401, and supported by the Austrian FFG, Bridge Project No. 871261. We thank A1 Telekom Austria AG for their support.

References

[1] P. Gomes, C. Olaverri-Monreal, and M. Ferreira, “Making Vehicles Transparent Through V2V Video Streaming,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 930–938, Jun. 2012.

[2] M. Laner, P. Svoboda, E. Hasenleithner, and M. Rupp, “Dissecting 3G Uplink Delay by Measuring in an Operational HSPA Network,” in *Passive and Active Measurement*, Atlanta, GA, USA, 2011, pp. 52–61.

[3] M. Laner, P. Svoboda, P. Romirer-Maierhofer, N. Nikaein, F. Ricciato, and M. Rupp, “A comparison between one-way delays in operating HSPA and LTE networks,” in *2012 10th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, Paderborn, Germany, May 2012, pp. 286–292.

[4] P. Svoboda, M. Laner, J. Fabini, M. Rupp, and F. Ricciato, “Packet delay measurements in reactive IP networks,” *IEEE Instrumentation Measurement Magazine*, vol. 15, no. 6, pp. 36–44, Dec. 2012.

[5] M. Laner, P. Svoboda, S. Schwarz, and M. Rupp, “Users in cells: A data traffic analysis,” in *2012 IEEE Wireless Communications and Networking Conference (WCNC)*, Shanghai, China, Apr. 2012, pp. 3063–3068.

[6] V. Raida, P. Svoboda, and M. Rupp, “Constant Rate Ultra Short Probing (CRUSP) Measurements in LTE Networks,” in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, Chicago, IL, USA, Aug. 2018, pp. 1–5.

[7] A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*. Prentice Hall Press, 2010.

[8] Ookla, LLC., “Speedtest by Ookla - The Global Broadband Speed Test,” http://www.speedtest.net/, visited on 2020-01-30.

[9] G. Aceto, F. Palumbo, V. Persico, and A. Pescapé, “Available Bandwidth vs. Achievable Throughput Measurements in 4G Mobile Networks,” in *2018 14th International Conference on Network and Service Management (CNSM)*, Rome, Italy, Nov. 2018, pp. 125–133.

[10] J. Strauss, D. Katabi, and F. Kaashoek, “A Measurement Study of Available Bandwidth Estimation Tools,” in *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, Miami Beach, FL, USA, 2003, pp. 39–44.

[11] Y. Xu, Z. Wang, W. K. Leong, and B. Leong, “An End-to-End Measurement Study of Modern Cellular Data Networks,” in *Proceedings of the 15th International Conference on Passive and Active Measurement - Volume 8362*, Los Angeles, CA, USA, 2014, pp. 34–45.

[12] V. Ribeiro, R. Riedi, J. Navrátil, and L. Cottrell, “pathChirp: Efficient Available Bandwidth Estimation for Network Paths,” *Proceedings of Passive and Active Measurement Workshop*, Apr. 2003.

[13] Mingzhe Li, M. Claypool, and R. Kinicki, “WBest: A bandwidth estimation tool for IEEE 802.11 wireless networks,” in *2008 33rd IEEE Conference on Local Computer Networks (LCN)*, Montreal, Que, Canada, Oct. 2008, pp. 374–381.

[14] A. K. Paul, A. Tachibana, and T. Hasegawa, “NEXT-FIT: Available Bandwidth Measurement over 4G/LTE Networks – A Curve-Fitting Approach,” in *2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA)*, Crans-Montana, Switzerland, Mar. 2016, pp. 25–32.

[15] T. Oshiba, K. Nogami, K. Nihei, and K. Satoda, “Robust available bandwidth estimation against dynamic behavior of packet scheduler in operational LTE networks,” in *2016 IEEE Symposium on Computers and Communication (ISCC)*, Messina, Italy, Jun. 2016, pp. 1276–1283.

[16] M. Fowler, “UserStory,” Apr. 2013, https://martinfowler.com/bliki/UserStory.html, visited on 2019-10-01.

[17] J. Lewis and M. Fowler, “Microservices,” Apr. 2014, https://martinfowler.com/articles/microservices.html, visited on 2020-01-31.

[18] ESnet, “iPerf3: A TCP, UDP, and SCTP network bandwidth measurement tool,” https://github.com/esnet/iperf, visited on 2019-11-12.

[19] V. Raida, P. Svoboda, M. Kruschke, and M. Rupp, “Constant Rate Ultra Short Probing (CRUSP): Measurements in Live LTE Networks,” in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, Shanghai, China, May 2019, pp. 1–6.