Hercules: An Emulation-Based Framework for Transport Layer Measurements over 5G Wireless Networks

Andrea Pinto[‡], Andrew Ashdown[*], Tanzil Bin Hassan[*], Hai Cheng[*],
Flavio Esposito[‡], Leonardo Bonati[*], Salvatore D'Oro[*], Tommaso Melodia[*],
Francesco Restuccia[*]
‡ Department of Computer Science, Saint Louis University, USA
* Institute for the Wireless Internet of Things, Northeastern University, USA

ABSTRACT

The adoption of Next-Generation cellular networks is rapidly increasing, together with their achievable throughput and their latency demands. Optimizing existing transport protocols for such networks is challenging, as the wireless channel becomes critical for performance and reliability studies. The performance assessment of transport protocols for wireless networks has mostly relied on simulation-based environments. While providing valuable insights, such studies are influenced by the simulator's specific settings. Employing more advanced and flexible methods for collecting and analyzing end-to-end transport layer datasets in realistic wireless environments is crucial to the design, implementation and evaluation of transport protocols that are effective when employed in real-world 5G networks. We present Hercules, a containerized 5G standalone framework that collects data employing the OpenAirInterface 5G protocol stack. We illustrate its potential with an initial transport layer and 5G stack measurement campaign on the Colosseum wireless network testbed. In addition, we present preliminary post-processing results from testing various TCP Congestion Control techniques over multiple wireless channels.

CCS CONCEPTS

• **Networks** → **Transport protocols**; **Network measurement**.

ACM Reference Format:
Andrea Pinto, Andrew Ashdown, Tanzil Bin Hassan, Hai Cheng, Flavio Esposito, Leonardo Bonati, Salvatore D'Oro, Tommaso Melodia, Francesco Restuccia. 2023. Hercules: An Emulation-Based Framework for Transport Layer Measurements over 5G Wireless Networks. In *The 17th ACM Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization 2023 (WiNTECH '23), October 6, 2023, Madrid, Spain.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3615453.3616516

1 INTRODUCTION

The transition of Internet transport layer protocol algorithms like Transmission Control Protocol (TCP)—originally designed and tailored to wired networks—to the fast-evolving 5G networks is an extremely delicate task. Such a transition, however, is not straightforward and comes with many challenges that stem largely from the absence of interaction between the higher-layer Congestion Control (CC) protocol and the Physical (PHY)/Medium Access Control (MAC) functionalities. Despite the fact that the design of new CC algorithms for wireless networks is currently an active and hot topic in the literature [2, 10, 14, 21], testing these solutions in real-world scenarios can be costly and time-consuming. Indeed, getting access to production-grade 5G deployments is hard (if not impossible) to obtain. For this reason, researchers have resorted to 5G simulation tools, such as ns-3 [20] to address this issue. However, performance measured through these tools results in approximations due to the inherent limitations of the simulation process. Specifically, the obtained simulation results deviate—in some cases even substantially [19]—from real-world experimental measurements [11]. For instance, simulations generally rely upon statistical models that are not able to accurately capture the complex wireless phenomena that radios experience in real-world environments. As a result, there is a strong need for 5G emulation environments that can provide researchers with a more accurate, realistic, and reliable platform to conduct CC-related research and obtain results that tightly match the behavior experienced in the real world.

WiNTECH '23 , October 6, 2023, Madrid, Spain

ACM ISBN 979-8-4007-0340-9/23/10…$15.00
https://doi.org/10.1145/3615453.3616516

Existing Challenges. The evolution and utilization of TCP over 5G networks are facing new challenges. The quality of wireless channels is highly variable, and it heavily affects user performance metrics such as throughput and latency. Conventional congestion control algorithms, which base their decisions on predefined criteria such as packet loss or delay, often struggle to adapt their behavior in such complex and changing environments. Therefore, there is a need for innovative congestion control mechanisms that can effectively handle the unique characteristics of 5G networks and optimize TCP performance accordingly [14].

As a consequence, the research community has deemed it necessary to design novel solutions that adjust TCP parameters by jointly considering transport-layer and PHY/MAC-layer metrics [12, 21, 24]. A variety of simulation tools to test new CC algorithms have been proposed in the literature [3, 23]. However, experimental, hardware-based tools are less readily available as radio devices are expensive, and hardware deployment, configuration, and maintenance are complex tasks that require expertise. In addition to the need for experimental platforms to conduct large-scale and repeatable CC experiments, there is also a need for tools that enable extensive data collection capabilities where metrics inherent to the PHY and MAC layers of the wireless protocol stack can be combined and analyzed alongside those available at the transport layer, so as to capture the effects of the wireless channel on CC algorithms. As a result, it is clear that emulation environments used for the design and evaluation of TCP CC need to provide PHY/MAC layer metric logging functionalities, together with the ability to experiment with diverse and heterogeneous Radio-frequency (RF) scenarios that exhibit unique wireless channel characteristics. Indeed, such emulation frameworks can provide researchers with the ability to analyze and quantify the effectiveness of existing and newly designed CC algorithms against diverse RF conditions, offering the tools to identify inefficiencies and design flaws and to gather insights that can help in the development of CC algorithms that are more effective and robust against changing channel conditions.

Summary of Contributions. To address the above challenges and limitations, in this paper, we propose Hercules , a novel full-stack framework that (i) enables end-to-end TCP CC experiments with data collection capabilities spanning PHY, MAC, and transport layers; (ii) provides the ability to instantiate an end-to-end 3GPP-compliant 5G protocol stack via OpenAirInterface (OAI) [18]; and (iii) combines the high-fidelity RF emulation capabilities of the Colosseum network emulator [5] to perform testing and evaluation under a variety of RF scenarios and wireless conditions. Our code is available with an open-source license at [1]. Hercules allows researchers to perform reproducible CC experiments in a controlled yet realistic 5G emulation environment, thus facilitating the design of 5G-tailored TCP algorithms. As such, this research contributes to the improvement of CC solutions that can make 5G networks more efficient and reliable. Moreover, we demonstrate Hercules's capabilities and provide some initial results using different TCP CC algorithms.

The rest of the paper is organized as follows: Section 2 outlines the related work in the domain of TCP CC for 5G networks. Section 3 describes Hercules's components and functionalities. Section 4 showcases the experimental, data collection, and analysis capabilities of Hercules. Section 5 discusses open problems, and Section 6 concludes our work.

2 RELATED WORK

This section outlines the current work and challenges relating to the need for CC protocol evaluation systems for 5G. We report on the main emulation results in the literature and discuss the pros and cons of the different approaches.

A comprehensive survey of TCP CC is provided in [14], where the authors examine related work describing the state-of-the-art for TCP CC in 5G networks. The implementation of different CC algorithms in 5G networks faces various challenges, including blockage, beam misalignment, frequent handovers, inadequate buffer sizes, interference from non-data signals, and changes in the data flow. The survey discusses these challenges and their potential impact on the TCP CC performance, and points out research directions such as latency vs. throughput trade-offs, queuing delay and bufferbloat, and flow fairness. Inspired by these challenges, we aim at investigating transport layer measurements and flow fairness over real 5G wireless channels, following the metrics analysis approach of [8].

Simulations are extensively evaluated over ns-3 mmWave simulation module as in [15] to measure the performance of multiple TCP CC algorithms in mmWave networks. Their work extensively evaluates various channel conditions within a simulation environment. However, it leaves open the question of how to transition these simulation results to a more realistic testbed where the results can be verified and further developed using actual waveforms and hardware devices.

As an improvement, to test transport protocol layers over actual wireless traces, the authors of [3] propose Zeus as an improved version of the Mahimahi link emulator [17]. The authors conducted throughput and delay comparisons of multiple CC protocols over various 5G channel traces, including real-life cellular 5G connections, simulated mmWave 5G channels, and WiGig (802.11ad) communications. As this work uses the Mahimahi emulator, the authors reduce the network to a single end-to-end bottleneck based on their collected traces, making it difficult to evaluate all the spectrum of the wireless connection parameters.

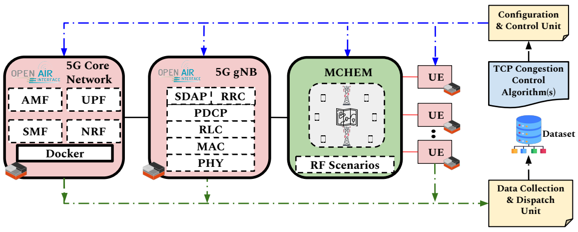


Figure 1: The Hercules architecture is composed of a Configuration and Control unit to configure transport and wireless layer experiments, the Data Collection and Dispatch Unit to collect such metrics at the different levels of the stack, a set of OAI containers, and the Radio Frequency channel emulation.

While trace-driven wireless simulations are valuable, they may lack reliability as they as they do not necessarily incorporate hardware devices for testing and evaluation. Such techniques only represent specific network environments and may not provide the variety that is needed for CC analysis in 5G networks. The high variability of network conditions in 5G deployments poses challenges for accurate modeling using software simulators like ns-3 or pre-recorded traces that do not incorporate hardware components. Furthermore, trace-driven and software simulations may not fully capture the complex interactions between transport layer protocols and the physical layer, which are essential for optimizing network performance. Consequently, the impact of different optimization techniques or changes in the network configuration may not be accurately reflected in purely trace-driven simulations.

To ensure a more comprehensive evaluation of the Internet protocols over 5G networks, our Hercules framework offers an emulation environment with a more accurate representation of the wireless system compared to the current literature, allowing researchers to gain deeper insights into the network behavior. We offer a set of tools to collect and analyze datasets to facilitate the analysis of transport metrics over wireless metrics. We will open source Hercules upon the acceptance of this paper.

3 ARCHITECTURE OVERVIEW

The main building blocks of Hercules are illustrated in Figure 1. They consist of OpenAirInterface 5G containers, a Configuration and Control Unit (CCU), and a Data Collection and Dispatch Unit (DDU). As the evaluation results are achieved on top of the Colosseum testbed, we use its Massive Channel Emulator (MCHEM) to emulate the radio signals. We describe the components individually in this section.

3.1 Configuration and Control Unit (CCU)

This unit offers an interface to configure and control the target experiment. The CCU is a set of configuration files and scripts to select the TCP CC algorithm, experiment type i.e., Uplink (UL) or Downlink (DL), experiment duration, and RAN parameters (e.g., sub-carrier spacing, carrier frequency, numerologies, etc.). With this unit, we empower the user to select any existing or customized TCP CC algorithm that is added to the kernel, in conjunction with the required 5G radio settings. To generate TCP traffic, in each experiment, we use `iperf3` [6], a traffic generator tool designed to test network performance through the use of transport-layer protocols. Our experiments show how we used this unit to test the connection between the core network and the User Equipment (UE) over the 5G wireless channel. The user can define the duration of the session and collect the experimental data using our Data Collection and Dispatch unit (DDU) described below.

3.2 Data Collection and Dispatch Unit

The Data Collection and Dispatch Unit (DDU) collects transport layer data from the traffic flow and MAC layer data from the 5G protocol stack during each experiment. Upon completing the data collection phase, the DDU provides a set of data analytics tools to parse the raw data, convert it into structured `CSV` files that contain the evolution of the different metrics over time, and generate plots to visualize data distribution, statistics, and temporal behavior.

5G Wireless Link Dataset: The 5G link-related data and metrics we collect in our framework are extracted directly from the gNodeB (gNB) by gathering approximately 30 parameters and metrics related to the PHY/MAC layers of the protocol stack for each UE. Among others, these include UE reported data such as Reference Signal Received Power (RSRP), Signal to Noise Ratio (SNR), UL and DL throughput, Modulation and Coding Scheme (MCS) index values, and Block Error Rate (BLER). As mentioned earlier, this has been made possible by extending the OAI gNB software stack to enable a fast and exhaustive data extraction. The above metrics are individually captured for each UE and stored in a `CSV` file at the gNB to facilitate post-processing and fine-grained data analytics. The gNB reports data to the DDU with a granularity of one second and each entry in the `CSV` file is tagged with a timestamp to indicate the capture time.

End-to-end Dataset: The transport layer end-to-end metrics are collected using the `tcpdump` packet analyzer.[7] TCP traffic flow is captured in `PCAP` format, and the `ss` [13] utility tool is used to access the TCP CC information every 20ms. Together with the 5G wireless metrics, these tools can facilitate a better understanding of how TCP CC algorithms and end-to-end performance metrics are impacted due to the dynamic nature of the 5G channel.

Once the data collection phase has been completed, the DDU provides a set of scripts to parse the collected transport layer raw data and generate a set of end-to-end metrics. These include the sending rate, throughput, fairness, retransmission rate, Round-trip time (RTT), inflight packets, and Congestion Window (CWND) parameters. We use the `dpkt`[22] module to extract the information from the packet `PCAP` captures. Specifically, in UL, the transmission rate is computed directly from the `PCAP` file recorded at the UE terminal, while the throughput is calculated at the destination (i.e., the core) utilizing the `PCAP` file recorded at the core network; the reverse is true for the DL scenario. We also compute Jain's index [9], which is generally used as a fairness indicator to show how fairly the communication channel is shared among the different TCP flows. Note that Internet traffic in the context of cellular networks is competing only for flows going to the same UE. For each arriving acknowledgment, we also calculate the RTT over a time interval for each given TCP timestamp. To calculate the actual packet loss, we count the retransmissions of TCP segments before the bottleneck. The inflight data metric represents the number of bytes that have been sent but have not yet been acknowledged. It is calculated by subtracting the maximum observed sequence number and acknowledgment number in the capture prior to reaching the 5G channel. Similar to the 5G wireless link dataset, the DDU also produces a `CSV` file for the end-to-end dataset containing timestamps and values for each of the above metrics, so as to guarantee that the two datasets are consistent and aligned in time.

Finally, to simplify and streamline data visualization, users can use the DDU to generate plots of each metric over the experiment time interval, as we will discuss in Section 4.

3.3 Radio Channel Emulation

Our framework is evaluated on the Colosseum testbed by using its Standard Radio Nodes (SRNs) and a Massive Channel Emulator (MCHEM) [5]. Each SRN consists of a 48-core Intel Xeon server, a NVIDIA GPU and controls a USRP X310 Software Defined Radio (SDR). Users of Colosseum can reserve and remotely access these resources to carry out experiments on the system. The MCHEM, which includes 128 USRP X310, allows users to faithfully emulate a variety of realistic wireless channels through a network of Field Programmable Gate Arrays (FPGAs) on ATCA-3671 boards. The emulation is performed through the use of *RF scenarios*, which consist of a time-varying set of 4 non-zero complex-valued channel taps. These represent how the channel between every pair of nodes in the experiment evolves over time. Channel taps are fed at run-time to the MCHEM, emulating the channel response each node would experience if operating in the considered real-world environment. Hence, through their use, scenarios can capture and reproduce wireless phenomena such as fading, mobility, and multi-path.

At the time of the experiment, the wireless waveforms transmitted by the SDRs are fed to the MCHEM, which processes them to apply the channel conditions specified through the selected RF scenario. Additionally, users can create their own RF scenarios to evaluate their solutions at scale in controlled but realistic environments [4]. Relevant to this paper, we mention [4], where the authors have developed and released to the community several RF scenarios tailored to cellular networking applications. These scenarios emulate cellular deployments in several cities across the world and capture important aspects such as varying UE mobility, topologies, and distribution of the UE. Each scenario is 600 seconds long and can be looped continuously during experimentation. Our framework harnesses these scenarios to analyze the effects of varying channel conditions on user TCP CC performance.

3.4 OAI 5G Deployment

The pink shaded blocks in Figure 1 represent the Linux Containers (LXCs) wherein OAI is installed and hosted.[1] OAI offers an open-source 3GPP-compliant 5G Radio Access Network (RAN) and core. Our framework utilizes two LXC images (OAI-CORE and OAI-RAN) to provide an end-to-end 5G Standalone (SA) architecture to instantiate and operate a standard-compliant 5G network on Colosseum. Our containers are based on OAI version 2023.w16. These LXC images are derived from [16] and have been extended to integrate data collection functionalities that are necessary to extract MAC layer metrics from the gNB. The 5G SA core LXC (OAI-CORE) hosts a dockerized Access and Mobility Management Function (AMF), User Plane Function (UPF), Session Management Function (SMF), and Network Repository Function (NRF). This is a minimalist deployment of the 5G core network supported by OAI. The OAI-RAN LXC can operate either as a gNB or as a UE depending on the experimental requirements.

Within Colosseum, LXCs are instantiated on the SRN compute nodes, which can be accessed remotely throughout the execution of the experiment. The SRN hosting the gNB (i.e., the OAI-RAN LXC container) is connected to the SRN hosting the 5G core (i.e., the OAI-CORE LXC container) via the internal network infrastructure of Colosseum. The SRNs hosting the UE containers interface to the gNB through the wireless channels emulated by MCHEM. Although OAI supports both FR1 and FR2 bands, as of today, the hardware infrastructure of Colosseum supports FR1 bands only. Therefore, in this paper, we limit our focus to such bands.

---

[1] LXC is a well-established containerization technology that makes it possible to run multiple isolated Linux systems on a single host machine.

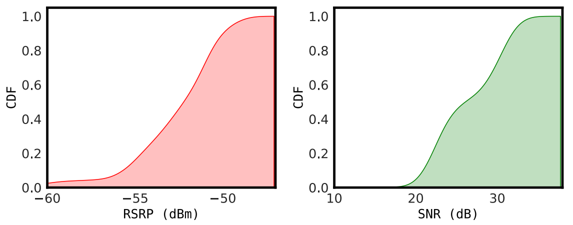
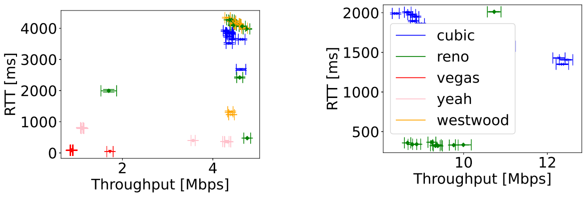
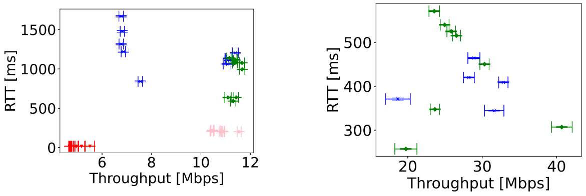


Figure 2: CDF of RSRP and SNR observed in the utilized portion of the Rome moderate mobility scenario.



(a) Uplink 24 PRB

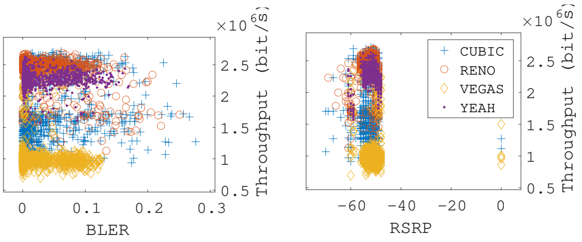
(c) Uplink 106 PRB



(b) Downlink 24 PRB

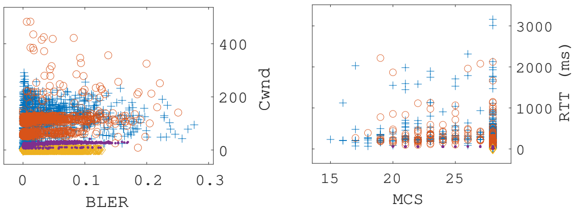
(d) Downlink 106 PRB

Figure 3: RTT vs. throughput for different congestion control algorithms in DL and UL, and for different PRBs. Different congestion control algorithms can be selected to study their performance over Hercules.



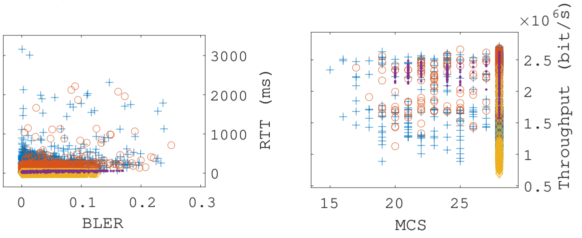
(a) BLER vs throughput

(d) RSRP vs througuput



(b) BLER vs CWND

(e) MCS index vs RTT



(c) BLER vs RTT

(f) MCS index vs throughput

Figure 4: TCP congestion control algorithm comparison for 24 PRB, DL traffic.

4 HERCULES EVALUATION

In this section, we showcase the capabilities of Hercules. Specifically, we present the results from two distinct experiments where we examine and compare the behaviors of several congestion control algorithms in realistic wireless scenarios over the Colosseum testbed.

4.1 TCP Congestion Control over 5G

Experimental setup: To analyze and compare how different TCP CC algorithms react to varying wireless conditions, we first consider a relatively simple yet effective experiment with one UE only connected to the core with a single TCP flow. For each experiment, the DDU is configured to perform data collection for an interval of 100 seconds. We analyze and compare the behavior of a variety of TCP CC algorithms under UL and DL traffic conditions as well as different numerologies (i.e., 24 and 106 Physical Resource Blocks (PRBs) corresponding to a channel bandwidth of 10MHz and 40MHz, respectively). Each CC algorithm has been tested individually on the same RF scenario to perform a fair comparison, and each experiment was run ten times. We use the Colosseum Rome RF scenario [4] to emulate channel conditions with moderate UE mobility (i.e., 3 m/s). The RSRP and SNR Cumulative Distribution Functions (CDFs) corresponding to the considered scenario are shown in Figure 2. During the first 100 seconds of this scenario (which is the temporal portion of the scenario we utilized for our experiments), the RSRP generally varies from -60 dBm to -45 dBm, while the SNR is most often between 18 dB and 38 dB.

Experimental results: Figure 3 reports RTT and throughput for each independent run using confidence intervals. In all cases, the individual TCP CC algorithms show different behavior for both the 24 and 106 PRB cases, as well as UL and DL cases. Figure 3b shows that `CUBIC`, `YeAH`, and `Reno` are achieving high throughput, but only `YeAH` exhibits a low RTT. Consequently, `YeAH` appears to be the best choice for a DL, single flow with 24 PRBs (for this RF scenario). Instead, if the goal is to prioritize lower RTT values (i.e., low end-to-end latency) over high throughput, Vegas would be the best choice. In UL, Figure 3a shows that the CC algorithms appear to deliver lower throughput and RTT values, but generally with high variance. Vegas is the best choice to achieve low latency, while YeAH can generally deliver high throughput at the price of a slight increase in RTT. For the 106 PRB case we show CUBIC and Reno as shown in Figure 3c and Figure 3d.

To identify relationships and coupling between transport layer end-to-end metrics and the 5G metrics, in Figure 4 we report the data points collected during our experiments and show how metrics at the MAC layer relate to those collected at the transport layer for the case of 24 PRBs and DL traffic. This analysis is relevant as it shows how metrics that capture lower layer conditions (e.g., RSRP, MCS index, and BLER) might impact end-to-end metrics and parameters such as RTT, throughput and CWND. As previously observed in Figure 3, Figures 4a and 4c show that Vegas achieves the lowest throughput and RTT; similarly, it also experiences the lowest BLER values, as shown in Figure 4a. Although CUBIC generally reaches higher throughput values and a larger CWND size than Vegas, (Figures 4a and 4b), we notice that CUBIC's RTT (and even BLER) are (generally) higher than the other CCs (Figure 4c). We also notice that CUBIC experiences a slightly larger dynamic range in MCS index, which also seems to affect the RTT (Figure 4e). On the contrary, Figure 4e shows that the other CC algorithms have a relatively more predictable MCS index selection. Moreover, recall that all experiments share the same RF scenario and channel conditions. This is shown in Figure 4d where the RSRP for all configurations takes values in the same range but results in different throughput values depending on the specific CC algorithm being used.

4.2 TCP channel contention over 5G

In this analysis, we consider the same configuration as the previous section but aim to examine the coexistence of multiple TCP connections over both UL and DL 5G connections. To achieve our goal, we consider the case where the UE establishes two simultaneous iperf3 sessions that use different TCP CC algorithms. Specifically, the first session uses CUBIC while the second uses Reno. The DDU is configured to run tcpdump at both the UE and core sides to capture network traffic, while the ss tool is used at the iperf3 client side to capture the TCP CC socket statistics.

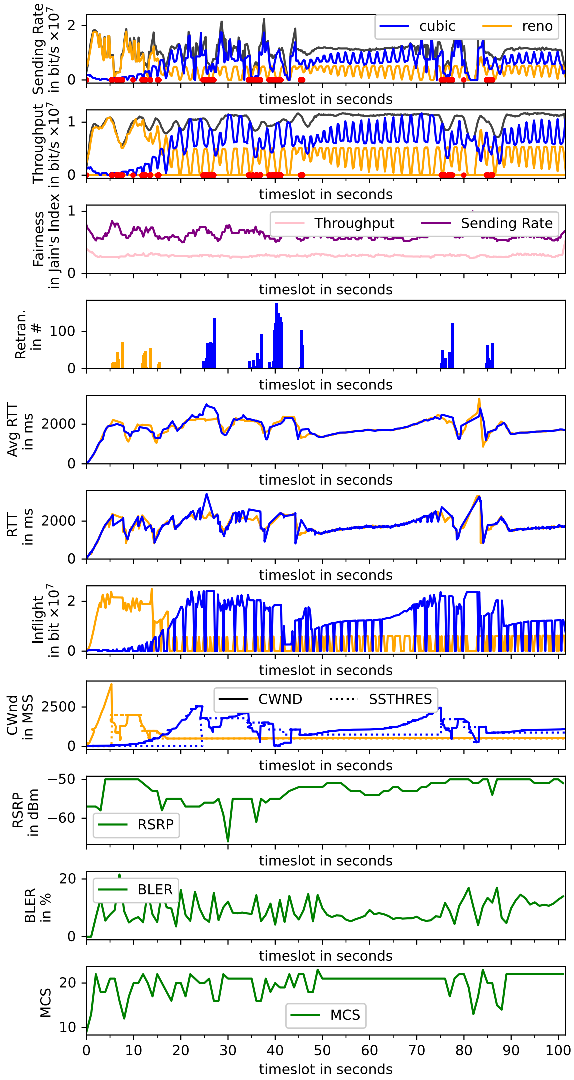


Figure 5: Downlink analysis of a single UE. Two distinct flows using Reno and CUBIC are established on the connection with the 5G core. The black curve represents the total bandwidth achieved in the first two plots, while the red dots indicate retransmissions. BLER, MCS, and RSRP are shown as a representation of MAC layer metrics.

Figure 5 summarizes the evolution of relevant metrics over time at different protocol stack layers for the DL case. In contrast to the results presented in the previous section, in this case, the two TCP flows compete with each other to access the bandwidth available over the 5G wireless channel. Specifically, Figure 5 illustrates the competition between Reno and CUBIC for channel occupancy. The sending rate and throughput metrics show that Reno occupies a significant portion of the available bandwidth during the first 15 seconds. This reflects on the initial inflight data of the Reno flow which appears higher than that of the CUBIC flow. However, Reno promptly adjusts its sending rate in response to congestion signals, resulting in lower throughput but potentially achieving better fairness, here measured in terms of Jain's index for both the sending rate and the actual throughput at the destination. In both cases, the flows do not demonstrate fair values, highlighting the presence of unfairness in their competition. This behavior aligns with the expected characteristics of the Reno algorithm. On the other hand, `CUBIC` constantly tries to adapt its CWND to the channel conditions, which are described by the RSRP and MCS index plots. For example, between seconds 50 and 75, we see that `CUBIC` increases the CWND to make the most out of the better channel conditions until, at around 80s, the BLER increases and more retransmissions occur. Indeed, `CUBIC` exhibits a more aggressive behavior by utilizing most of the available bandwidth, potentially leading to higher throughput. However, this comes at the cost of decreased fairness compared to Reno. We also notice that there are two main fluctuations in the RSRP, BLER, and MCS index occurring in the intervals $[20, 40]$s and $[75, 90]$s. This behavior seems to reflect to an oscillation of the transport-layer metrics, causing a drop in the total throughput, more retransmissions at the transport-layer, fluctuation of the `CUBIC` CWND value, and higher inflight data for the `CUBIC` connection. The RTT is similar for both CC algorithms. This metric strongly depends on the quality of the channel, whose fluctuations seem to result in retransmissions for both CC algorithms. Figure 6 presents a similar analysis but focuses on the UL traffic. Reno dominates the first half of the experiment in this case, while `CUBIC` increases the CWND adapting to the channel. After the first 50 seconds, Reno decreases its sending rate, which gives `CUBIC` the chance to increase the sending rate and improve its throughput. As no retransmissions occurred during this experiment, we have omitted its corresponding plot. The RTT exhibits a similar behavior for both algorithms. Furthermore, Reno demonstrates a more stable inflight packet rate and CWND handling compared to `CUBIC` in this case.

5 DISCUSSION AND OPEN PROBLEMS

The Transport Protocol analysis software is tailored for evaluating TCP CC measurements. Our emphasis was on showcasing wireless link experiments. Even if measurements were done in a multi-hop setup, our framework would work, as it uses input traffic traces from both source and destination. In our evaluations, the wireless link was the only network bottleneck. In a complete path scenario, Figures 5-6 In our evaluations, the wireless link was the only network bottleneck. Hardware and software limitations on the Colosseum testbed, combined with OAI constraints, restricted our experiment's throughput. Setting up a custom 5G software stack on a local testbed could be considered. Note that Colosseum can scale the experiments to multiple UEs. Similarly,

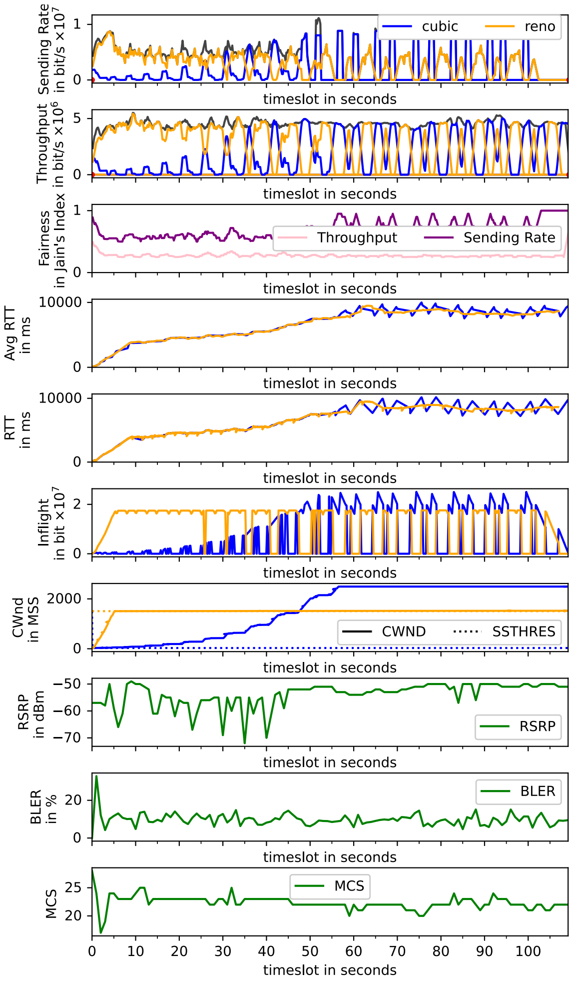


Figure 6: Uplink analysis of a single UE. Two distinct flows using Reno and `CUBIC` are established on the connection with the 5G core. In the first two plots, the black curve represents the total achieved bandwidth, while the red dots indicate instances of retransmission.

our ability to evaluate a fraction of the TCP CC algorithms was limited by Colosseum's kernel version, which does not currently support the latest CC algorithms.

6 CONCLUSION

The 5G network is expected to achieve high performance in terms of throughput and delay. As the current transport layer protocols have been designed for wired settings, it is nontrivial to understand how they adapt to 5G networks. Most of the current research has relied heavily upon simulators to test experimental solutions and measure the performances of the TCP Congestion Control protocols. While 5G CC simulation studies are valuable, there is a need to go beyond these approaches if TCP CC algorithms are to be optimized for realistic 5G network conditions. Thus, to have greater confidence in the accuracy of CC studies for 5G, there is a need for realistic and reliable 5G emulation environments for TCP CC experimentation.

We propose Hercules, a framework to allow the research community to collect new datasets over specific settings of interest. We have harnessed the capabilities of Northeastern University's Colosseum network emulator along with the OAI 5G stack to create an effective 5G emulation testbed with wireless and end-to-end data collection/analysis capabilities and realistic RF channel characteristics. Moreover, Hercules provides analysis tools to parse the collected data. We present two main experiments and explain how we achieved the results with our framework, providing a discussion of the results. We highlight the differences between some of the various CC protocols' behavior over 5G UL and DL settings. Moreover, we describe a scenario of competition between two different CC algorithms.

ACKNOWLEDGMENTS

This work has been funded by NSF grants OAC-2201536, CNS-2134973, CNS-2120447, CNS-1925601, and ECCS-2229472, by the Air Force Office of Scientific Research under contract number FA9550-23-1-0261, and by the ONR under award number N00014-23-1-2221.

REFERENCES

[1] 2023. Hercules: An Emulation-Based Framework for Transport Layer Measurements over 5G Wireless Networks. https://https://github.com/pintauroo/5G_measurement_framework.

[2] Aziza Alzadjali, Flavio Esposito, and Jitender Deogun. 2020. A Contextual Bi-armed Bandit Approach for MPTCP Path Management in Heterogeneous LTE and WiFi Edge Networks. In *The Fifth ACM/IEEE Symposium on Edge Computing*.

[3] Rohail Asim, Muhammad Khan, Luis Diez, Shiva Iyer, Ramon Aguero, Lakshmi Subramanian, and Yasir Zaki. 2022. ZEUS: An Experimental Toolkit for Evaluating Congestion Control Algorithms in 5G Environments. *arXiv preprint arXiv:2208.13985* (2022).

[4] Leonardo Bonati, Salvatore D'Oro, Stefano Basagni, and Tommaso Melodia. 2021. SCOPE: An Open and Softwarized Prototyping Platform for NextG Systems. In *Proc. of ACM Intl. Conf. on Mobile Systems, Applications, and Services (MobiSys)*. Virtual Conference.

[5] Leonardo Bonati, Pedram Johari, Michele Polese, Salvatore D'Oro, Subhramoy Mohanti, Miead Tehrani-Moayyed, Davide Villa, Shweta Shrivastava, Chinenye Tassie, Kurt Yoder, et al. 2021. Colosseum: Large-Scale Wireless Experimentation Through Hardware-in-the-Loop Network Emulation. In *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*. IEEE, 105–113.

[6] Jon Dugan, John Estabrook, Jim Ferbuson, Andrew Gallatin, Mark Gates, Kevin Gibbs, Stephen Hemminger, Nathan Jones, Feng Qin, Gerrit Renker, Ajay Tirumala, and Alex Warshavsky. 2023. iPerf - The ultimate speed test tool for TCP, UDP and SCTP. https://iperf.fr/

[7] The Tcpdump Group. Accessed 2023. TCPDUMP 4. https://github.com/the-tcpdump-group/tcpdump

[8] Benedikt Jaeger, Dominik Scholz, Daniel Raumer, Fabien Geyer, and Georg Carle. 2019. Reproducible measurements of TCP BBR congestion control. *Computer Communications* 144 (05 2019).

[9] R. Jain, D. Chiu, and W. Hawe. 1998. A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems. arXiv:cs/9809099 [cs.NI]

[10] Nathan Jay, Noga Rotman, Brighten Godfrey, Michael Schapira, and Aviv Tamar. 2019. A deep reinforcement learning perspective on internet congestion control. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*. PMLR, 3050–3059.

[11] Glenn Judd and Peter Steenkiste. 2004. Repeatable and Realistic Wireless Experimentation through Physical Emulation. *SIGCOMM Comput. Commun. Rev.* 34, 1 (January 2004), 63–68.

[12] Madhan Raj Kanagarathinam, Sukhdeep Singh, Irlanki Sandeep, Hanseok Kim, Mukesh Kumar Maheshwari, Jaehyun Hwang, Abhishek Roy, and Navrati Saxena. 2020. NexGen D-TCP: Next Generation Dynamic TCP Congestion Control Algorithm. *IEEE Access* 8 (2020), 164482–164496.

[13] Alexey Kuznetsov. Accessed 2023. Socket Statistic tool. https://man7.org/linux/man-pages/man8/ss.8.html

[14] Josip Lorincz, Zvonimir Klarin, and Julije Ožegović. 2021. A Comprehensive Overview of TCP Congestion Control in 5G Networks: Research Challenges and Future Perspectives. *Sensors* 21, 13 (June 2021), 4510.

[15] Pablo Jimenez Mateo, Claudio Fiandrino, and Joerg Widmer. 2019. Analysis of TCP Performance in 5G mm-Wave Mobile Networks. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. IEEE, Shanghai, China, 1–7.

[16] Eugenio Moro, Gabriele Gemmi, Michele Polese, Leonardo Maccari, Antonio Capone, and Tommaso Melodia. 2023. Toward Open Integrated Access and Backhaul with O-RAN. arXiv:2305.06048 [cs.NI]

[17] Ravi Netravali, Anirudh Sivaraman, Keith Winstein, Somak Das, Ameesh Goyal, and Hari Balakrishnan. 2014. Mahimahi: a lightweight toolkit for reproducible web measurement. In *Proceedings of the 2014 ACM conference on SIGCOMM*. ACM, Chicago Illinois USA, 129–130.

[18] Navid Nikaein, Mahesh K. Marina, Saravana Manickam, Alex Dawson, Raymond Knopp, and Christian Bonnet. 2014. OpenAirInterface: A Flexible Platform for 5G Research. *SIGCOMM Comput. Commun. Rev.* 44, 5 (oct 2014), 33–38.

[19] Monika Prakash and Atef Abdrabou. 2020. On the Fidelity of NS-3 Simulations of Wireless Multipath TCP Connections. *Sensors* 20, 24 (2020).

[20] George F. Riley and Thomas R. Henderson. 2010. The ns-3 Network Simulator. In *Modeling and Tools for Network Simulation*, Klaus Wehrle, Mesut Günes, and James Gross (Eds.). Springer, 15–34.

[21] Alessio Sacco, Flocco Matteo, Flavio Esposito, and Guido Marchetto. 2021. Owl: Congestion Control with Partially Invisible Networks via Reinforcement Learning. In *IEEE Conference on Computer Communications (INFOCOM)*. 1–10.

[22] Dug Song. Accessed 2023. dpkt. https://dpkt.readthedocs.io/en/latest/index.html

[23] Menglei Zhang, Marco Mezzavilla, Russell Ford, Sundeep Rangan, Shivendra Panwar, Evangelos Mellios, Di Kong, Andrew Nix, and Michele Zorzi. 2016. Transport layer performance in 5G mmWave cellular. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, San Francisco, CA, USA, 730–735.

[24] Ticao Zhang and Shiwen Mao. 2020. Machine Learning for End-to-End Congestion Control. *IEEE Communications Magazine* 58, 6 (2020), 52–57.