Measuring One-Way Delay in Real 5G Scenarios

Andreas Ingo Grohmann*, Mauri Seidel*, Leonardo Badia†, Marie-Theres Suer‡, Oscar Dario Ramos-Cantor‡
Sebastian A. W. Itting§, Frank Hofmann‡, and Frank H.P. Fitzek*§
*Deutsche Telekom Chair of Communication Networks, Technische Universität Dresden, Germany,
Email: {Andreas_Ingo.Grohmann, Mauri.Seidel}@tu-dresden.de
†*Dept. of Information Engineering* University of Padova, Italy, leonardo.badia@unipd.it
‡*Advanced Distributed Intelligence* Robert Bosch GmbH, Hildesheim, Germany,
Email: {Marie-Theres.Suer, OscarDario.RamosCantor, Frank.Hofmann2}@de.bosch.com
§Centre for Tactile Internet with Human-in-the-Loop (CeTI), Email: {Sebastian.Itting, Frank.Fitzek}@tu-dresden.de

***Abstract*—Measurements of one-way delay in 5G networks are essential for evaluating system performance in various practical scenarios. This paper presents a comprehensive study comparing three different packet generators on real-world deployments and their associated challenges. We explore different topologies, types of terminals, and network congestion conditions, discussing the suitability and limitations of different packet generators. Moreover, we propose practical considerations to overcome limitations in capturing dynamic and heterogeneous 5G network environments and provide insights for system designers in precisely measuring and characterizing one-way delay in 5G networks for real-world scenarios.**

***Index Terms*—5G Non-Public Network; Performance evaluation; One Way Delay; Measurements; Industry 4.0.**

I. Introduction

One of the key performance metrics for the upcoming Fifth Generation (5G) of cellular networks is one-way delay (OWD), i.e., the time it takes for a packet to travel from a source to a destination in a network [1]. Low OWD is essential for many industrial applications, such as remote control and automated manufacturing [2]. However, it is affected by a number of factors, such as the network architecture, the traffic load, and the distance between the source and destination [3].

Non-Public 5G networks can be set up in a variety of configurations [4] depending on the actual use case. In this work, we set up a 5G carrier-grade network at Bosch Corporate Research in an industry environment in Hildesheim, Germany. The testbed used for our measurements consists of two main parts, namely, the 5G network and the user equipments (UEs). For the latter, we used up to five *Askey NDQ 1300* devices, connected to Intel NUCs. The former is a full stack carrier-grade 5G network, including two routers, the indoor radio unit, the base band unit, the power supply, and two multi purpose servers. After verifying the functionality, we tried different configurations of the 5G network and performed a measurement campaign based on different data rates, number of UE devices and cross traffic. We use a gambit of measurement tools to collect OWD data under different conditions.

In particular, the focus of our analysis is the comparison of different packet generation tools, for which we compared state-of-the-art scriptable generators. In contrast to existing publications [5], we compared packet generators in the context of real world 5G over-the-air measurements instead of clean lab environments. Besides the performance evaluation, our first-hand report from the comparison of the tools can be useful for practitioners in the field. In acknowledgment of the limitations within this study, it is noted that the inclusion of related work is insufficient for a comprehensive overview. This omission was a deliberate decision to conserve space, thereby allowing for an extended analysis of the packet generator comparison and the incorporation of preliminary results from industrial 5G measurements. For an extensive review of related work, readers can refer to [6], which utilizes a comparable testbed and provides a thorough examination of existing literature.

Our findings show that OWD in 5G non-public networks (NPNs) is generally in the range of milliseconds, but can increase significantly when the network is under heavy load. Our analysis provides valuable insights into the factors that affect OWD in 5G NPNs and can serve to evaluate the usability of 5G connectivity for many kinds of use cases.

The remainder of this paper is organized as follows. In Section II, we describe the testbed we set up for our measurement campaign. In Section III, we show a detailed comparison on different software packet generators. Section IV, presents the results of the measurement campaign along with the interpretation and we conclude in Section V.

II. Setup

A. User Equipment

5G UE devices use a variety of different modems to connect to 5G networks. For our testing, we used the *Quectel RM500Q-GL*, which leverages the *Qualcomm X55* chipset. It is a high-performance modem that supports 5G New Radio (NR) and *LTE Cat.18*. In theory, it can achieve download speeds of up to 2.5 Gbps and upload speeds of up to 900 Mbps. It is also power efficient, making it ideal for integrated devices as small x86 based PC we used it in. We used this setup for the first measurements with only one user in the network.

To scale up our testing, while maintaining homogeneous devices, we had to move to a different UE setup. For this purpose, we acquired multiple *Askey NDQ 1300* devices, connecting each of them to a x86 based computer. The *Askey NDQ 1300* is a 5G NR USB dongle supporting download speeds up to 2.7 Gbps and upload speeds up to 800 Mbps. The Askey devices are utilizing a *Qualcomm X55* chipset, the same as the

979-8-3503-6836-9/25/$31.00 ©2025 IEEE

Quectel modem. Due to licensing restrictions in the baseband, we were only able to use five of the *Askey NDQ 1300* devices simultaneously. During verification testing we used a *FritzBox 6850* and a *Nokia 8.3 5G* as well, equipped with a *Qualcomm X55* and *Qualcomm X52* modem, respectively.

B. 5G Network

The deployed 5G network consists of a carrier-grade commercially available off-the-shelf (COTS) 5G radio access network (RAN), *Open5GS v2.6.6* 5G core network (CN), and a COTS server hosting the multi-access edge cloud (MEC). The utilized 5G NPN is responsible for providing wireless connectivity to the UEs in the testbed. The base station consists of a high capacity mobile backhaul router, an indoor radio unit with one *Radio Dot* and a baseband.

The router is a high-performance router that offers a variety of features that make it ideal for use in a private 5G network. It has twelve $1\,$Gbps and eight $10\,$Gbps Ethernet ports, which provide a wide range of connectivity options for devices in the network. The router supports precision time protocol (PTP) and *SyncE* for time synchronization, to ensure that UEs and the MEC in the network are synchronized. Within the context of this work, we synchronized all endpoints in the network to perform OWD measurements. Overall, the router is a powerful platform that can be used to create a high-performance 5G NPN. The baseband does the RAN processing and is therefore crucial to create a 5G NPN. Based on the 5G standard, it offers a number of features like high capacity and advanced connectivity that make it ideal for industrial applications.

The indoor radio unit in combination with the *Radio Dot* takes care of the radio transmissions for indoor deployments. It is a purpose-built, high-performing device for single- or multi-operator deployments in medium-to-large venues such as airports, manufacturing halls, offices, and stadiums. It can serve up to eight *Radio Dots* from one centralized location, with a $90\,$m Ethernet reach. The *Radio Dot* is a small, low-power device that is used to provide the coverage of a 5G network. Typically mounted on walls or ceilings, it can create a seamless and reliable wireless connection. Within the scope of this project, one *Radio Dot* was more than sufficient to provide good coverage for the whole testbed.

Due to software license restrictions, we are not able to change the 5G RAN configuration. The available licenses allowed $100\,$MHz bandwidth and 30 kHz sub carrier spacing (SCS), resulting in a slot duration of $0.5\,$ms. Within the software, two different time division duplex (TDD) patterns are possible. All tested UEs only supported the pattern $7/3$, i.e., seven downlink (DL) slots followed by three uplink (UL) slots. The transmission power was configurable on the baseband, but even the lowest level ensured good service for the whole test area. The maximal modulation and coding scheme is *256 QAM*.

In addition to the RAN components, the utilized 5G network also comprises two versatile multi-purpose servers. One of these servers hosts the 5G CN, while the other is dedicated to management tasks. Both servers possess sufficient capabilities to support MEC as well. In the context of this work, the server not hosting the CN hosted the packet generators. The 5G CN assumes the responsibility of routing traffic, managing subscriptions, and providing security services. For this work, the chosen 5G CN was *Open5GS v2.6.6* [7], an open-source solution that offers comparable performance to commercial alternatives. Preliminary work [8] has shown that *Open5GS* is capable enough even on less powerful hardware. The CN was running locally on the server that was also running the measurement tools.

C. Packet Generators

MoonGen [9] is a high-speed, scriptable packet generator that can be used to test network devices and applications. It is an open-source software, written by a research group at TU Munich and is available on GitHub [10]. The packet generator is written in Lua and uses the Data Plane Development Kit (DPDK) to provide high performance. It can generate traffic at line rate on a single CPU core and can scale to multiple cores to generate even more traffic. *MoonGen* can generate a variety of traffic types, including TCP, UDP, and ICMP. It can also generate traffic with different packet sizes, inter packet delays, and burst sizes. *MoonGen* was used in previous work [1], that we are going to extend in the scope of this work.

Rude&Crude [11] is a packet generator tdeveloped in the early 2000s with its latest release in 2013. It is an open-source software available on Sourceforge. *Rude&Crude* is written in C and sends packets directly to the network interface (NIF). This allows *Rude&Crude* to generate traffic at very high speeds. It can also generate different packet sizes, inter packet delays, and burst sizes.

udp_ping is a versatile packet generator developed by the Deutsche Telekom Chair of Communication Networks. It is written in Python using the *asyncio* library. Originally, it was developed to be a tool like the well-known ping, but sending UDP packets instead of ICMP. Over time, it has been adapted to different use cases and can now also send one-way traffic recorded at the remote host, using a small C program to mirror the packets, *udp_ping* is a flexible and easy-to-read/modify tool, which can generate different packet sizes and rates.

III. Comparison of Generators

We start with a comprehensive comparison of the three packet generators, with specific focus on their precision for the delay measurements at different inter packet delays. Table I presents an overview of the advertised features of each tool. *MoonGen* claims to achieve the highest possible performance without any compromise. This leads to some disadvantages, regarding the deployment flexibility and usability. *MoonGen* requires exclusive access to the NIF used, blocks at least one whole CPU core per NIF and is limited to a few kernel versions and certain NIF. The same instance of *MoonGen* that generates the packets also has to receive them, to allow the most precise measurement. Hence, there needs to be a loop in the setup of *MoonGen*. *Rude&Crude* and *udp_ping* act like any other program without special requirements. In contrast,

TABLE I
FEATURE COMPARISON OF THE THREE SOFTWARE PACKET GENERATORS

Feature	**MoonGen**	**Rude&Crude**	**udp_ping**
Language	Lua/C/C++	C	Python/C
Packet types	TCP, UDP, ICMP	UDP	UDP
Rates	Variable $>$ 100 Gbps	Variable $>$ 1 Gbps	Variable $>$ 1 Gbps
Performance	High	Medium	Medium/Low
Software stack	DPDK based	OS kernel based	OS kernel based
Ease of use	Difficult	Medium	Easy
Documentation	Minimal	Minimal	Sufficiant
License	MIT	GPLv2	internal
In this work			
Setup	Sender/Receiver same machine	Sender/Receiver different PC	Sender/Receiver same/different PC
Remote PC	Forward at sys level	Receiver	Forward at app level / Receiver
Requirements	Special network card/driver	PTP sync	Nothing / PTP sync
Required port	2 exclusive network ports	shared port	shared port

the measurement precision of the user space packet generators *Rude&Crude* or *udp_ping* is bound by the synchronization quality of the end points. We configure the router as PTP grandmaster utilizing the *G8275.1* profile. The resulting synchronization precision between the different end points is in the range of double-digit nanoseconds.

Despite of the different approaches, we deploy a setup as close as possible for all packet generators to achieve comparable results. Therefore, *MoonGen* runs on the server hosting the CN. In the DL, it sends packets via the 5G network to the UE. The UE PC forwards the packets after reception back to the CN server using a wired connection. For the UL, it is the other way round. The user space tools work in a one way configuration, i.e., they run on the CN server and send their packets via the 5G network to the UE, where they terminate. The UE PC is synchronized via a wired connection with PTP.

Both setups have their drawbacks. In the case of *MoonGen*, the wired link and the forwarding in the UE introduced delays independent of the 5G link, which we aimed to measure. These delays are typically around a few hundred microseconds, with minimal jitter. On the other hand, with *Rude&Crude* and *udp_ping*, additional delays are introduced during the forwarding process from the NIF up to the application running in the user space of the PC. These delays, ranging in the few hundred microseconds, are present at both the sender and the receiver's side. Notably, these delays are heavily influenced by the system load. Yet, compared to our focus on a single digit millisecond range, all those delays are negligible.

After considering these theoretical aspects, we conduct measurements to compare the performance of the tools in over-the-air 5G transmissions. Fig. 1 is a boxplot providing an overivew of the measured inter-packet times for different configurations, comparing *udp_ping*, *Rude&Crude*, and *MoonGen*. The inter packet delays configured for all three tools were set at 0.1 ms, and 1 ms. On the left, the measured inter packet delay when using *udp_ping* is shown to almost never hit the configured value, indicating low accuracy. The measured delay was not constant as well, which can be seen by the pure size of the boxes, reflecting poor precision. For both delays, the measured inter-packet time is significantly higher than what configured.

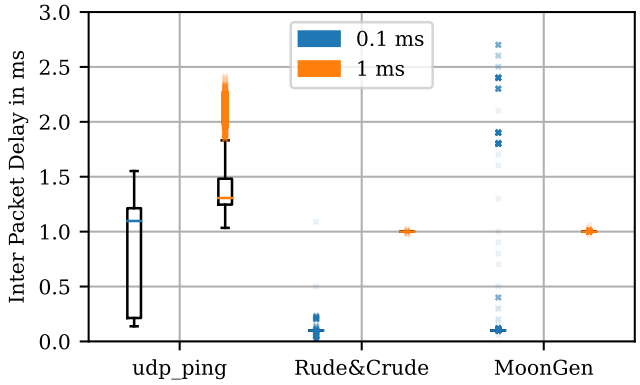

Fig. 1. Measured inter packet delay for *udp_ping* vs. *Rude&Crude* vs. *MoonGen*, for configured values of 0.1 ms and 1 ms

When using *Rude&Crude*, the measured results align closely with the configured values. For both inter packet delays the boxes are small, reflecting a high precision and accurate at the configured value. For this setting, the majority of the measured values still fall within the configured time, but some instances experience significant delays, even exceeding 3 ms.

Fig. 1 also compares the actual inter-packet delays of *MoonGen* and *Rude&Crude*. Overall, in this case *Rude&Crude* and *MoonGen* perform about the same. This can be reasoned by the low amount of sent traffic. *MoonGen* has advantages by generating traffic in the order of hundreds of Gbps. As the used 5G network limits the data rate to a value lower than one Gbps, those advantages are not relevant for this work.

This comparison shows a big deviation in the inter packet delay, when using *udp_ping* compared to the other two packet generators. To save space and avoid visual clutter *udp_ping* is not used for the remainder of this work. As those results show no significant difference in measured values for *MoonGen* or *Rude&Crude*, yet *MoonGen* requires two exclusive ports for each running instance, which does not scale with the number of UEs based on hardware limitations, whereas we use *Rude&Crude*, which does not have such a requirement, for all upcoming measurements.

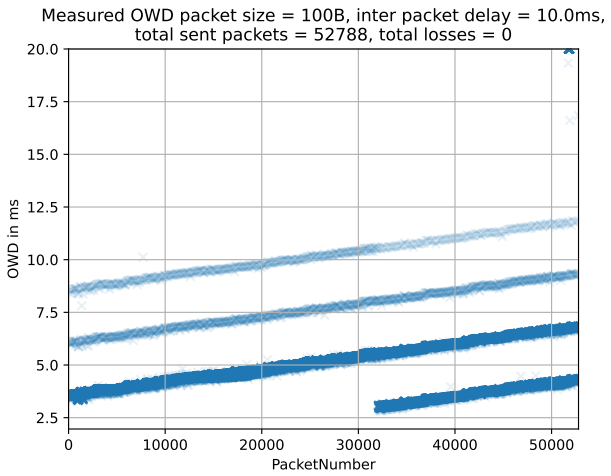

Fig. 2. OWD of single packets, sent with 10 ms inter-packet delay

As a next step, we measured the OWD using the 5G system. The result can be found in Fig. 2 and shows four discrete possible delay values for each packet, rather than a normal distribution. Those four discrete steps are visible, starting from about 5 ms with 2.5 ms distance to each next one. They are reasoned by the 5G frame structure [12]. If the inter packet delay would be perfectly aligned with the slots in 5G, the lines would be horizontal. However, in reality *Rude&Crude* sends a packet every 10.01 ms on average, instead of every 10.00 ms. As this does not align with the frame structure the lines are not perfectly horizontal, but rise a bit over the amount of packets.

To further investigate the observation of the discrete values we decreased the inter packet delay and the amount of packets we plotted in one graph. We observed a sawtooth pattern, which can be found in Fig. 3. The distance between the extreme values is about 5-6 ms in most cases and the periodicity is about 5-6 ms as well. Although the overall latency is higher, if we further decrease the inter packet delay this effect stays about the same, as visible in Fig. 4. Another effect can be seen even higher latencies are common as well, which is reasoned by queuing within the base station.

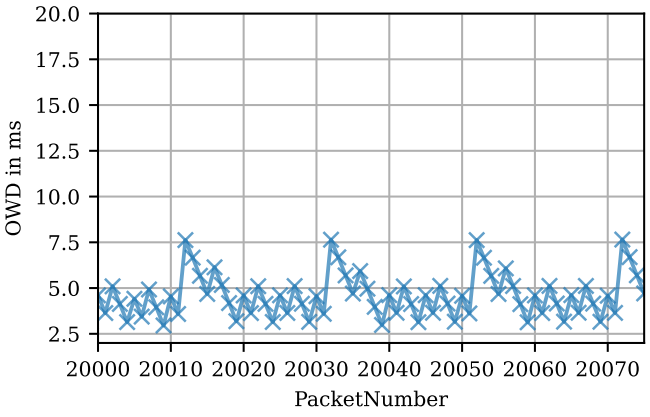

Fig. 3. OWD of single packets, sent with 1 ms inter-packet delay

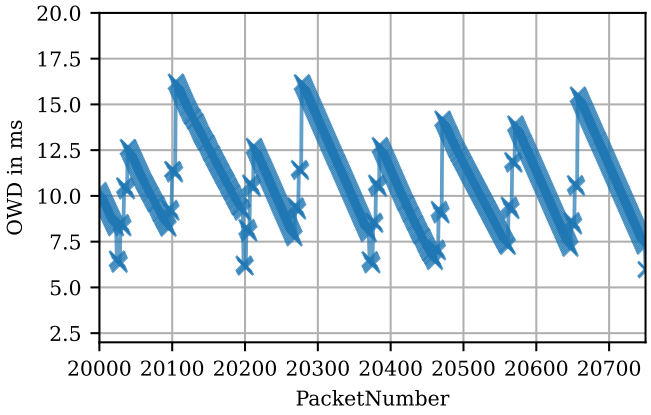

Fig. 4. OWD of single packets, sent with 0.1 ms inter-packet delay

IV. Measurements

This section describes measurements done by using up to five *Askey NDQ 1300* UE and the 5G network. Preliminary studies [1] showed that different packet sizes up to the maximum transmission unit (MTU) [13] have no impact on OWD; based on those findings, all measurements in this work use packets of 100 bytes. Measuring in the application layer, on top of multiple layers that include error correction [14], we did not face any packet losses.

A. One End Device

First, we used one UE for UL and DL traffic in parallel. Due to the fixed TDD pattern, UL transmissions do not influence DL and vice versa. The measurement lasted 60 s and the cumulative distribution function (CDF) of the experienced OWD can be found in Fig. 5. The figure shows a smaller delay in DL compared to UL in almost any case. In addition, decreasing the inter-packet delay from 10 ms to 0.1 ms slightly increases latency in both directions. As we are sending cyclic traffic and the inter packet delay is in the same range as the TDD patterns periodicity, the relative starting point might have an impact on the measurement.

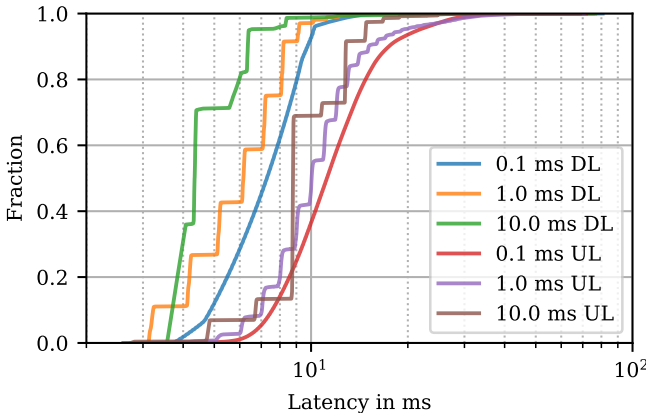

Fig. 5. Sending traffic with a single UE in the network for 60 s

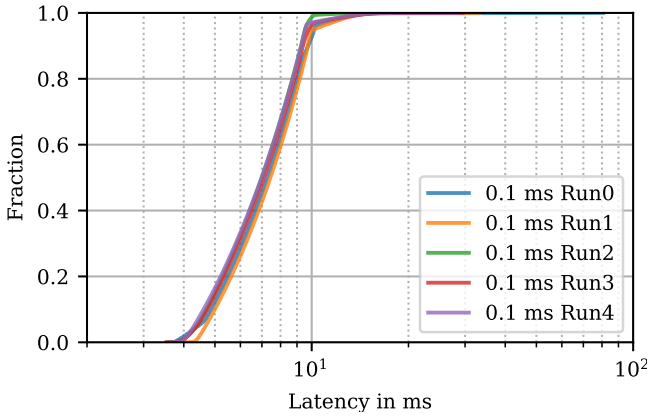

Fig. 6. Five different runs, one UE in DL with 0.1 ms inter-packet delay

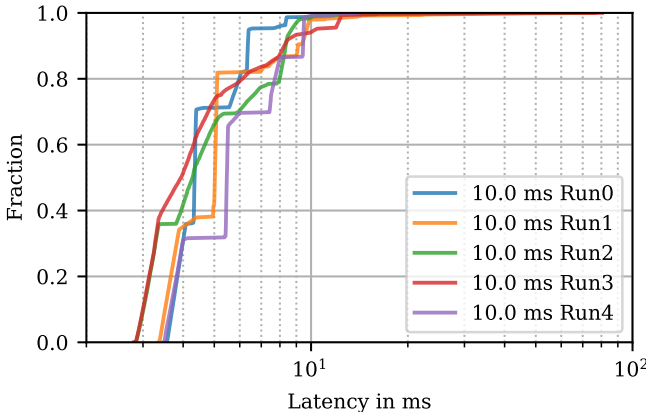

Fig. 8. Five different runs, one UE in DL with 10 ms inter-packet delay

To investigate the influence of the 5G frame structure [12] on our measurement, we took a closer look at each inter packet delay and to avoid confusion we limited the plots to DL. The results for 0.1 ms inter-packet delay are shown in Fig. 6. This figure shows no significant difference in all five runs. The runs deviate in runtime, where *Run0* and *Run1* took 60 s, *Run2* was 100 s, *Run3* 600 s and *Run4* 900 s. However, the results show no significant differences for 0.1 ms inter-packet delay. As the inter-packet delay is an order of magnitude smaller than a slot, this seems reasonable. The majority of packets arrived after 4-10 ms. This 6 ms spread can be due to some kind of timer in the UE, and we conjecture it may be inside the *Qualcomm X55* chip set. However, we did not find any documentation confirming this assumption.

Then, we increased the inter-packet delay to 1 ms, which resulted in the measured OWD shown in Fig. 7. This plot clearly demonstrates a distinction among the various runs. The runs exhibiting the smallest and largest OWD are approximately 1.5 ms apart. This discrepancy can be attributed to the slot configuration employed, as elucidated in Section II-B. Each TDD period consists of a sequence of slots with duration of 0.5 ms. Since a TDD pattern 7/3 is used, packets may need to wait for an additional time of up to 1.5 ms, depending on when they are created within the TDD pattern. This is almost exactly the distinction found between the runs in Fig. 7.

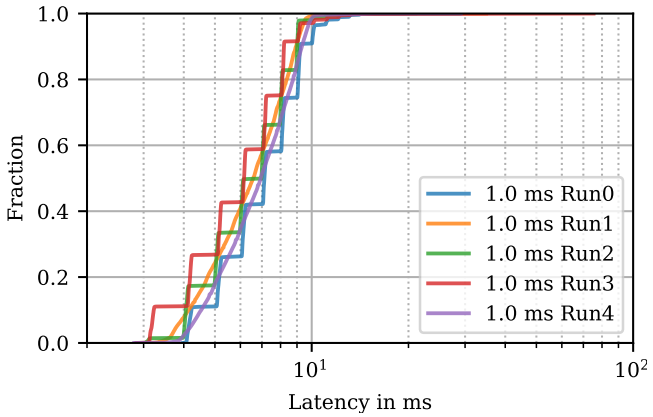

Fig. 7. Five different runs, one UE in DL with 1 ms inter-packet delay

Fig. 8 shows the same setup, but with 10 ms inter-packet delay. This plot shows about 1.5 ms distinction between individual runs as well for many cases. However, since only one packet was sent every 10 ms, there are two full TDD periods between two packets. This long time allows a variety of other effect to take place as well, which blurs the results.

B. Cross Traffic

After these investigations showing only one UE in the network, we focus on multiple UEs transmitting at the same time. To do so, we generated cross traffic using *IPerf* [15] on other UEs. *IPerf* created User Datagram Protocol (UDP) packets with a bitrate of 1 Gbps. We used one UE to transmit traffic in UL and another one for DL, both UEs trying to send as much data as possible, and we generated UL and DL traffic using *Rude&Crude*. Fig. 9 shows the result for DL, with and without cross traffic. The impact seems to be different depending on the inter-packet delay. For 1 ms inter-packet delay, the difference is smaller than the variation in Fig. 7. For 0.1 ms inter-packet delay, there is a noticeable impact as the mean delay increases from around 8 ms to 12 ms. This means that in the worst case the packets have to wait until the next TDD period, but get transmitted in that one for sure. For 10 ms the impact of cross traffic is the most significant according to Fig. 9, where some packets get up to 80 ms delay. Fig. 10 shows the UL direction of the same experiment. To present comparable graphs the x-scale is limited to 100 ms, regardless of the tail for 10 ms inter-packed delay with cross traffic going up to 300 ms. Fig. 10 shows an effect on cross traffic for all different inter-packed delays, even 1 ms.

V. Conclusions and Lessons Learned

We setup a 5G stand alone network using an indoor radio solution, connecting up to four kinds of devices and five devices of the same kind. We did a comprehensive comparison of three different software packet generators.

Concerning the packet generators, *udp_ping* gives an easy-to-read/use Python based approach, but it provides less ac-

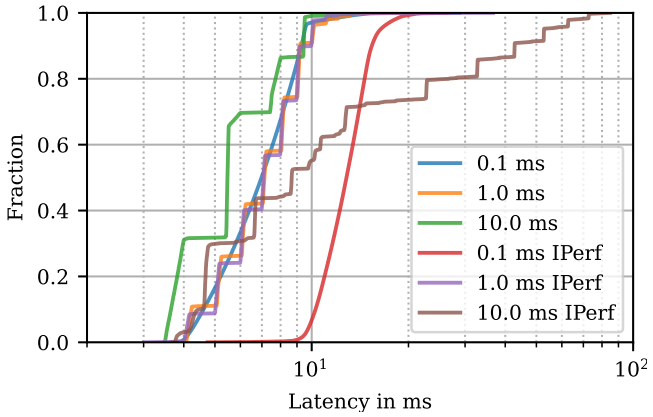

Fig. 9. One UE sending with cross traffic generated by *IPerf*, DL

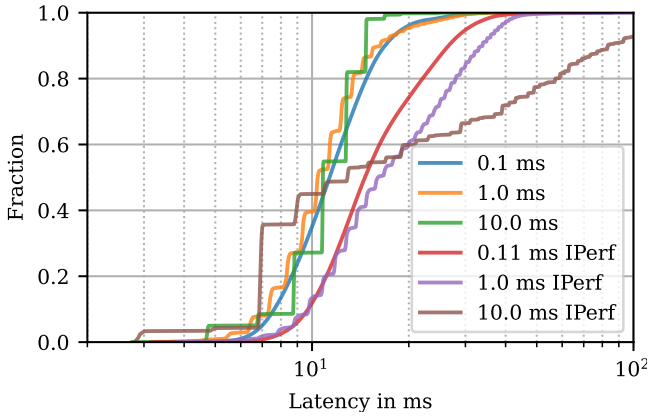

Fig. 10. One UE sending with cross traffic generated by *IPerf*, UL

curate packet generation and measurement precision. *MoonGen* requires exclusive hardware usage, which increases the requirements, without providing significant benefits compared to *Rude&Crude* when sending less than 1 Gbps of traffic. *Rude&Crude* is a publicly available open source generator, which gave the best user experience for our use case.

The primary objective was to transmit packets with a precision surpassing the expected OWD by at least two orders of magnitude. Using *Rude&Crude* as packet generator, we showed the capabilities of the implemented 5G network to obtain OWDs in the range of 5-10 ms in DL and 8-20 ms in UL. The measured delays were consistent for the duration of the measurement and the number of devices did not significantly impact the performance, the only limitations being given by the exhaustion of the licenses. We showed an effect on the OWD based on the time a packet was sent relative to the TDD pattern, with respect to the inter-packet delay. We presented recurring traffic patterns independent of the inter-packet delay, which indicate the existence of an up to 5 ms buffer in the UE.

Concerning other aspects of the implementation, we notice how a Radio-Dot can cover a large area, making it a good choice for applications that require wide coverage. Moreover, since UL and DL are separate within a TDD period, they are scheduled independently and do not influence each other.

These results appear to be heavily influenced by the specific implementation and especially the available licenses. Imagining a 5G network based on hardware of a less strict vendor or even generic hardware and open source software, a more comprehensive study would be possible. Network operators could adapt the configuration of the 5G network according to their applications needs. The comprehensive setup and verification process outlined herein can serve as a foundational blueprint for future 5G measurement configurations, particularly when comparing various RAN architectural options [6].

Acknowledgment

This work was supported by the German Federal Ministry of Economic Affairs and Climate Action (BMWK) as part of the joint French-German project 5G-Opera, grant 01MJ22008A; the Federal Ministry of Education and Research of Germany in the program “Souverän. Digital. Vernetzt.;” the Italian PRIN project 2022PNRR “DIGIT4CIRCLE,” project code P2022788KK; the Joint Project 6G-life, ID 16KISK001K; and the German Research Foundation (DFG) as part of Germany’s Excellence Strategy – EXC 2050/1 – Project ID 390696704 – Cluster of Excellence Centre for Tactile Internet with Human-in-the-Loop (CeTI) of Technische Universität Dresden.

References

[1] J. Rischke, P. Sossalla, S. Itting, F. H. Fitzek, and M. Reisslein, “5G campus networks: A first measurement study,” *IEEE Access*, vol. 9, pp. 121 786–121 803, 2021.
[2] L. Badia and A. Munari, “Status update scheduling in remote sensing under variable activation delay,” in *Proc. IEEE BalkanCom*, 2023.
[3] L. De Vito, S. Rapuano, and L. Tomaciello, “One-way delay measurement: State of the art,” *IEEE Trans. Instrum. Meas.*, vol. 57, no. 12, pp. 2742–2750, 2008.
[4] A. I. Grohmann, M. Seidel, C. Lehmann, T. Höschele, M. Reisslein, and F. H. Fitzek, “5G on the cheap: Configurable low-cost cellular industrial communication,” in *Proc. Int. Conf. Elec. Comp. Commun. Mechatron. Eng. (ICECCME)*, 2022, pp. 1–6.
[5] A. Botta, A. Dainotti, and A. Pescapè, “Do you trust your software-based traffic generator?” *IEEE Commun. Mag.*, vol. 48, no. 9, pp. 158–165, 2010.
[6] A. I. Grohmann, M. Seidel, S. A. Itting, R.-G. Cheng, M. Reisslein, and F. H. Fitzek, “Multi-UE 5G RAN measurements: A gamut of architectural options,” *IEEE Access*, vol. 13, pp. 1846–1866, 2024.
[7] S. Lee, “Open5GS,” https://github.com/open5gs/open5gs, [accessed 13-07-2024].
[8] G. Lando, L. A. F. Schierholt, M. P. Milesi, and J. A. Wickboldt, “Evaluating the performance of open source software implementations of the 5G network core,” in *Proc. IEEE/IFIP NOMS*, 2023, pp. 1–7.
[9] P. Emmerich, S. Gallenmüller, D. Raumer, F. Wohlfart, and G. Carle, “Moongen: A scriptable high-speed packet generator,” in *Proc. Internet Measur. Conf.*, 2015, pp. 275–287.
[10] P. Emmerich, “Moongen packet generator,” https://github.com/emmericp/MoonGen, [accessed 13-07-2024].
[11] J. Laine, S. Saaristo, and R. Prior, “Real-time UDP data emitter (rude) and collector for rude (crude),” https://sourceforge.net/projects/rude/, [accessed 13-07-2024].
[12] 3GPP, “Physical channels and modulation,” 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.211, 04 2022, version 17.1.0.
[13] D. S. E. Deering and J. Mogul, “Path MTU discovery,” RFC 1191, Nov. 1990. [Online]. Available: https://www.rfc-editor.org/info/rfc1191
[14] L. Badia, “On the effect of feedback errors in Markov models for SR ARQ packet delays,” in *Proc. IEEE Globecom*, 2009.
[15] R. McMahon, B. Kaushik, and T. Auckland, “Iperf: The TCP/UDP bandwidth measurement tool,” https://sourceforge.net/projects/iperf2/, 2005.