IEEE INFOCOM 2024 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) | 979-8-3503-8447-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/INFOCOMWKSHPS61880.2024.10620666

Evaluation of User Plane Function Implementations in Real-World 5G Networks

Sokratis Christakis*‡, Theodoros Tsourdinis*‡, Nikos Makris*†, Thanasis Korakis*† and Serge Fdida‡
*Dept. of Electrical and Computer Engineering, University of Thessaly, Greece
†Centre for Research and Technology Hellas, CERTH, Greece
‡Laboratoire LiP6/CNRS, Sorbonne University, Paris, France
Email: schristakis@uth.gr, ttsourdinis@uth.gr, nimakris@uth.gr, korakis@uth.gr, serge.fdida@lip6.fr

***Abstract*—5G technology, the latest advancement in mobile networks, promises increased data speeds, reduced latency, and enhanced device connectivity. Within the core network lies the User Plane Function (UPF), a critical component ensuring seamless data transfer to any data network, playing a major role in breaking out the traffic from the telecommunication network. The immense data forwarding demands on the UPF have sparked extensive research and technological innovations. This paper focuses on a system evaluation of different UPF implementations in a real-world 5G environment. Specifically, we deploy and test four distinct 5G networks based on SPGWU-UPF, P4-Switch-UPF, VPP-UPF, and SmartNIC-P4-UPF using the OpenAirInterface 5G framework. Through rigorous testing, we found that while the P4-UPFs outperformed in terms of throughput and packet loss, the VPP-UPF, a DPDK solution, showed promise in latency performance and cost-effectiveness, despite its higher CPU consumption. This study offers valuable insights into the operational challenges and benefits of various UPF implementations in real-world scenarios.**

***Index Terms*—5G, Edge Computing, Core Network, UPF, DPDK, P4, VPP**

I. Introduction

5G is the fifth generation of mobile networks and constitutes a technological development designed to offer increased data speeds, lower response times (known as latency), and increased connectivity for devices. It suggests an important step in the evolution of wireless communications and has many applications, including technologies that require huge amounts of data and minimal latency times (e.g. AR/VR, IoT devices, 4K/8K video streaming, etc.). To meet these demands, 5G uses an even more sophisticated network architecture than 4G, which lies on advanced and state-of-the-art technologies.

The 5G network can be divided into two main categories, the Radio Access Network (RAN) and the Core Network (CN). The RAN represents the wireless interface between the User Equipment (UE) and the antenna (gNB) that best serves it. On the other hand, the core network is responsible for operations such as registration and authentication of a user, security, and user tracking, in order to ensure a seamless transition between different gNBs and maintain a continuous and stable connection. Furthermore, it is responsible for forwarding the data received from the gNBs to the Internet. The network function that is dedicated to performing the latter operation is called the User Plane Function (UPF) and plays a crucial role in 5G networking, as every packet that reaches any data network is processed through it. The enormous amount of data that needs to be forwarded by the UPF function, has triggered the scientific community in researching and implementing advanced technologies for the best possible result.

In this work, we are evaluating different UPF implementations through repeated experiments, in a real environment under realistic conditions. More specifically, we use the 5G network provided by OpenAirInterface (OAI) and deploy four different 5G networks, which are based on a SPGWU-UPF, a P4-Switch-UPF, VPP-UPF and a SmartNIC-P4-UPF respectively. Evaluating in an actual 5G network environment is significant to provide real-world validity, ensuring that the results are applicable in practical scenarios. The insights gained from these evaluations extend to operational considerations, helping identify how easy or challenging it is to deploy, manage, and maintain each UPF implementation.

The rest of the paper is organized as follows. Section II contains information on research and studies relevant to this work. Sections III and IV include details on the system architecture and experimental findings. Finally, in Section V, we draw conclusions and outline some potential future work.

II. Related Work

5G introduces key changes to the core network by adopting a modular and cloud-based approach. A notable shift is the move from a traditional telecom monolithic setup (and closed-in vendor-specific hardware in many cases) to a Service-Based Architecture (SBA), which offers more flexibility and opportunities for network enhancements. Such innovation enables the network operators to deploy the network functions even as microservices in matching modern cloud-native technologies like Docker and Kubernetes [1]. In this way the network functions can be easily managed, scaled [2], and migrated [3] to cover the needs of the end-users. Furthermore, the softwarization of the network functions has opened the horizon for further novelties in the telecom architecture such as the Control and User plane separation (CUPS) [4]. This in turn brings new innovations mainly on the UPF side, which have the sole purpose of increasing bandwidth, reducing latency, adding flexibility and improving the Quality of Service (QoS).

The research leading to these results has received funding from the European Horizon 2020 Programme for research, technological development and demonstration under Grant Agreement Number No 101008468 (H2020 SLICES-SC). The European Union and its agencies are not liable or otherwise responsible for the contents of this document; its content reflects the view of its authors only.

979-8-3503-8447-5/24/$31.00 ©2024 IEEE

Multiple Access Edge Computing (MEC) takes advantage of such CUPS architectures by placing the user plane part of the UPF at the edge of the network, while the control plane side can run in the cloud [5]. Although MEC has enabled low-latency access to 5G networks suitable for ultra-reliable low-latency communication (URLLC) features, the bottleneck of performance still persists in the RAN configuration but also in the UPF implementation.

Focusing on the UPF implementations, several solutions are utilizing the Data Plane Development Kit (DPDK), a technology that contains a set of libraries and drivers for fast packet processing. One key advancement of the DPDK technology is that it is designed to bypass the traditional kernel-based network stack, allowing for fast packet processing directly in user space. By skipping the kernel's network stack, DPDK can achieve lower latency and higher throughput for packet processing [6]. To achieve high-performance UPF in [7] the authors proposed optimizations, including the use of DPDK for high throughput requirements. Their DPDK-based UPF solution achieved a performance of up to 40Gbps in a simulation environment. In [8] the authors highlighted the challenges of using DPDK in edge computing, particularly its CPU-intensive polling mechanism. They introduced an alternative in-kernel 5G UPF solution aligned with 3GPP Release 16, suitable for MEC deployments. This solution taps into the extended version of the Berkeley Packet Filter (eBPF)/eXpress Data Path (XDP), a kernel-based technology, showcasing its potential as a viable alternative to traditional DPDK methods. eBPF is a revolutionary technology within the Linux kernel that allows the execution of user-defined programs in a safe manner without changing kernel source code or loading modules [9]. XDP is a framework within the Linux kernel that leverages eBPF for high-performance packet processing directly at the device driver level before the kernel network stack gets involved [10] [11].

Several works also employ hardware solutions with the most promising one being the P4-driven UPF. P4 switches, especially when implemented in hardware, can process packets at a line rate, while maintaining the programmability of the flows, ensuring that the high throughput and low latency requirements of 5G networks are met. The authors in [12] showcase the performance benefits of using a P4 switch as a UPF and the implementation challenges. Additionally, in [13] the paper compares modern packet processing acceleration technologies, including P4, and their respective latency performance. The results highlight the advantages of hardware-based P4 implementations in terms of reduced tail latency.

In this work, we progress beyond existing literature by deploying a cloud-native end-to-end 5G network, integrating all the open-source UPF solutions in the deployed network, and evaluating them in a real-world setup. Specifically, we integrate all the softwarized UPFs, by starting with the legacy SPGWU, then moving to a DPDK-based solution named Vector Packet Processing (VPP) UPF, and finally utilizing a P4-Switch to function as a UPF. Moreover, as an alternative to DPDK-based solutions, we use the virtual interfaces provided by P4-SmartNICs through the single-root input/output virtualization (SRIOV) function. This capability bypasses the kernel stack and allows real-time processing of packets in user-space. On top, we deploy the UPF function and compare its performance with the rest of the solutions. Finally, we evaluate all the employed solutions in terms of throughput, latency, packet loss, and CPU consumption.

III. System Architecture

Our overall setup consists of two separate end-to-end 5G architectures; the softwarized UPF setup and the P4-Switch UPF setup. Fig. 1 illustrates both architectures. Both setups rely on OpenAirInterface's core network for the corresponding control network functions such as AMF, AUSF, SMF, UDR, and UDM. The way they differ is on the UPF implementation, since in the softwarized setup, the UPF is either spgwu-tiny or VPP, which are both software-based UPF solutions. The SmartNIC-UPF can be described by the softwarized setup as well, as the OAI architecture is the same, with the only exception that it uses the SRIOV interface provided by the SmartNIC-P4 card for the GPRS Tunneling Protocol (GTP) session. In contrast, the P4-Switch setup uses a hardware P4 switch to operate as a UPF. All network functions are deployed as microservices using Docker container runtime. To connect the end-users to the network we utilized the UERANSIM an open-source state-of-the-art 5G UE and RAN (gNodeB) simulator [14]. Each setup is deployed in real-world machines at our premises at LIP6-Sorbonne University and the NITOS testbed in Greece, which are both part of SLICES-RI([17]). The deployment specifications are summarized in Tables I, II. Below we analyze the respective UPF solutions that we integrated along with the various additional technologies. All implementations and each setup can be reproduced by following the instructions and deploying the codes. [1]

TABLE I: LIP6 Experimental Setup

System	Description
CPU	Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz
Cores	64
RAM	240GB
Operating System	Ubuntu 22.04.2 LTS
Linux Header Image	5.19.0-1022-lowlatency
MTU	1500
5G-Core Network	OpenAirInterface
UPF Types	SPGWU, VPP, SD-Fabric
P4 Switch	Wedge100BF-32QS
QFSP Cable	Mellanox 100Gb QFSP28
5G-RAN	UERANSIM
5G-UE	UERANSIM

A. SPGWU-Based UPF

The Serving and Packet Data Network Gateway User plane (SPGW-U) is a core element of 4G/LTE networks and is responsible for routing and forwarding user data packets.

[1]Information about each setup available at: https://github.com/schristakis/UPF-Evaluation

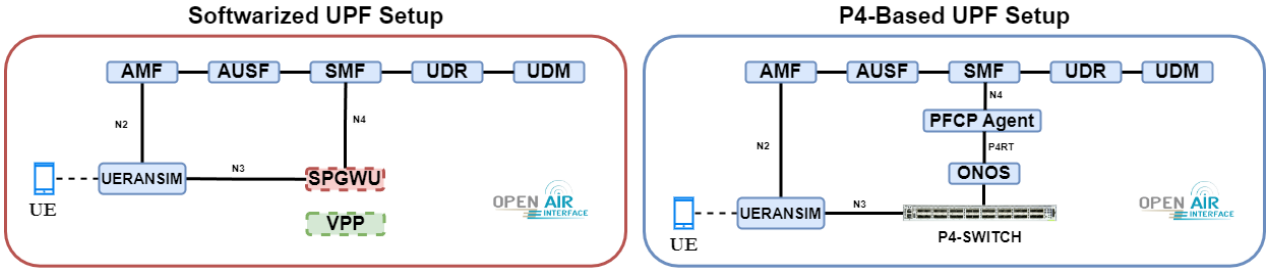


Fig. 1: Experimental Setup: Softwarized UPFs vs P4-Switch-based UPF

TABLE II: NITOS Experimental Setup

System	Description
CPU	11th Gen Intel(R) Core(TM) i7-11700K @ 3.60GHz
Cores	16
RAM	32GB
Operating System	Ubuntu 18.04.6 LTS
Linux Header Image	4.15.0-213-generic
MTU	1500
SmartNIC Type	Netronome Agilio CX 25GbE
5G-Core Network	OpenAirInterface
UPF Types	SmartNIC-P4 UPF
5G-RAN	UERANSIM
5G-UE	UERANSIM

Based on 3GPP cellular specifications, OpenAirInterface has developed 'oai-spgwu-tiny', which is an improved version of SPGW-U and initially was compatible only with 4G/LTE networks. The advent of 5G has triggered OAI to adapt its operation and integrate it into 5G networks as well. However, in many cases, this monolithic technical approach does not satisfy the high demands of 5G networks, and as a result, other UPF technologies are being considered.

B. VPP-Based UPF

Vector Packet Processing (VPP) is a high-speed packet-processing framework that can operate on a variety of CPUs. VPP differs from traditional packet processing as it processes multiple packets at once using a technique called vector processing, which utilizes the SIMD capabilities of modern processors to handle several packets in parallel. This approach increases the data plane performance, achieving high throughput and low latency that are comparable to, and sometimes surpass, specialized hardware solutions. VPP also leverages technologies like the Data Plane Development Kit (DPDK), which allows direct access to network interface cards (NICS), bypassing the operating system's networking stack, and reducing additional overhead. Furthermore, it allows the distribution of the packet processing workload across numerous CPU cores to optimize hardware usage and scale in overload conditions.

When compared to typical monolithic SPGWU systems in 5G networks, VPP provides numerous additional benefits. As a result, it has been utilized for User Plane Function solutions in 5G networks, to manage high-speed data traffic and low response times. Its high-performance packet processing capabilities, make it particularly suitable for the data-intensive and dynamic nature of 5G services. Additionally, VPP's adaptability allows for the rapid deployment of network slices, supporting the diverse and evolving use cases of 5G, from enhanced mobile broadband (eMBB) to ultra-reliable low latency communications (URLLC) and massive machine type communications (mMTC).

To integrate this solution we relied on the OpenAirInterface's implementation of VPP-UPF. This implementation utilizes a VPP User Plane Gateway (UPG) [15] that relies on the FD.io VPP plugin [16]. The user-plane functionalities comply with the 15th release of 3GPP.

C. P4-Switch-Based UPF

P4 is a programming language that enables packet switches to modify the way packets are processed, offering the flexibility to adapt to changing protocols and services, a key feature of 5G networks. The use of P4 upgrades packet switches from the limitations of fixed-mode hardware, allowing them to operate using reconfigurable rules based on the network's requirements. The ability to transform static hardware devices into real-time programmable devices inspired their usage and incorporation into UPF operations. This integration brings significant advantages as we can combine speed and fast hardware processing with programming flexibility.

Our P4-switch setup, which can be observed in Fig. 2, consists of several components, regarding Open Network Operating System (ONOS), management switches, Stratum and the Packet Forwarding Control Protocol (PFCP) Agent. ONOS is the network controller that manages the P4-Switch-based User Plane Function (P4-Switch UPF). It provides a platform for network configuration and administration, allowing for dynamic programming of network devices and real-time modifications to network policies. ONOS allows us to operate and control the setup, regardless of the underlying hardware, which is crucial for easy management and deployment. Furthermore, we utilize Stratum, a software that indicates the physical switches on how to handle the data that are being forwarded. We could consider Stratum as the component that translates the instructions received from ONOS, into specific, actionable commands for the hardware switch. PFCP Agent was developed to allow the control plane (i.e the SMF function) to communicate with and manage P4-programmable switches. More specifically, it is deployed to manage the communication between the SMF and the P4-Switch UPF functions, in order to successfully establish the PFCP session. As depicted in Fig. 2, ONOS, and OpenAirInterface functions are deployed on the

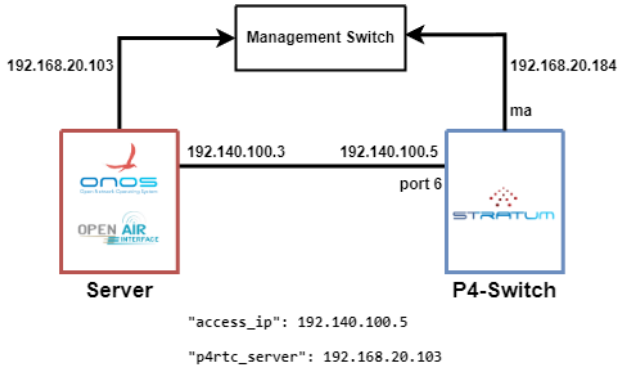


Fig. 2: P4-Switch-based UPF Experimental Architecture.

server, whereas the UPF and Statum are executed within the P4 switch itself. The server and the switch are interconnected and linked to a management switch, which serves as a centralized unit for administration and control. The *"p4rtc_server"* is the IP address that the P4-Switch UPF application (UP4-APP) is running on the ONOS controller. PFCP Agent communicates with ONOS (UP4-APP) via this IP, over the management network. For the user-plane network, we connected a QFSP cable from our server to the P4 port. The user-plane interface is the *"access_ip"* from the side of the P4 switch playing the role of the N3 interface. From the server side, the interface plays the role of the external data network (N6 interface).

D. SmartNIC-P4 UPF

Our setup for the SmartNIC-based SPGWU consists of two powerful machines provided by NITOS testbed. One is used to deploy the RAN functions and the other (which incorporates the SmartNIC-P4 card) is utilized to deploy the core functions of our 5G network. In order to provide low-latency access to the network functions, we use the virtual interfaces that are deployed along with the SRIOV function for the card and provide access directly to the packets written to the network in user-space. The card also supports programmable flows, which in our case are set to transfer the packets received on the physical port to the virtual port on top of which the UPF is instantiated. The general functionality remains the same as the one of legacy SPGWU, with the only exception that the GTP traffic is forced to go through the SmartNIC-P4 card's interface, to get the best performance possible (Fig. 3).

IV. Experimental Evaluation

To evaluate the performance of the various UPF types, we initially measured the throughput for both Uplink (UL) and Downlink (DL) traffic. Furthermore, we tested the performance of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) protocols, by utilizing a traffic generator tool (iperf) on the side of the UERANSIM and the external-data network and vice-versa. For a better and structured analysis, we divide the experimental findings into two subsections. Subsection IV-A describes the experiments that have been conducted in the real-world servers at our

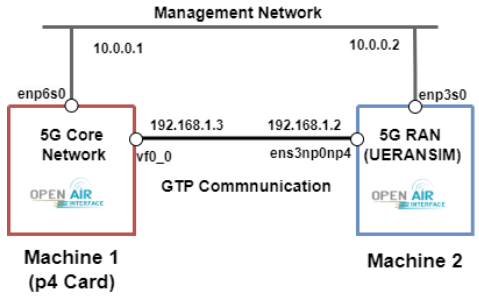
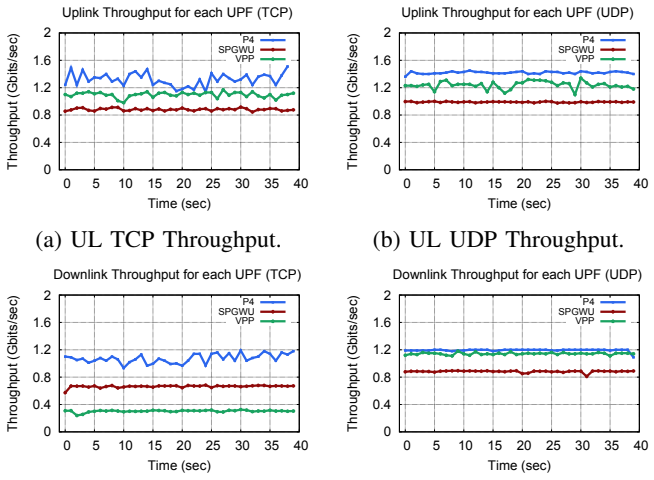


Fig. 3: SmartNIC-P4 UPF Experimental Architecture.

premises at LIP6-Sorbonne University, whereas Subsection IV-B is about our experiments in NITOS Testbed.

A. SPGWU-UPF, VPP-UPF, P4-SWITCH-UPF (LiP6/CNRS)

Fig. 4 illustrates all the throughput measurements for every UPF type. The P4-switch UPF achieved the highest throughput among the other solutions, having almost a 2-times higher throughput than the monolithic SPGWU. However, the VPP-based UPF being a software network function and indirectly a cheaper solution compared to the P4-switch UPF, reached relatively good performance, surpassing the SPGWU and approaching the performance of the P4-switch UPF. However, in the downlink TCP scenario, the VPP-UPF performed very poorly in comparison with the other conducted experiments.

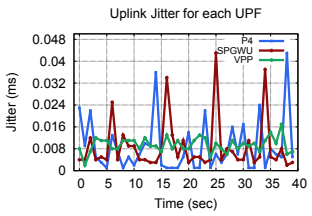


(a) UL TCP Throughput. (b) UL UDP Throughput.

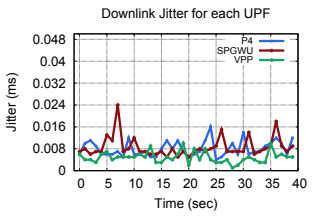
(c) DL TCP Throughput. (d) DL UDP Throughput.

Fig. 4: Throughput Evaluation: Uplink and Downlink.

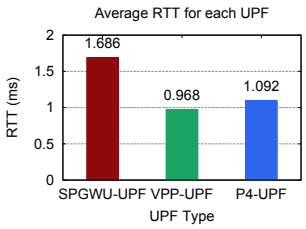
Additionally, the VPP-UPF offered the smoothest experience to the end-users in terms of latency. This can be observed in Fig. 5, which aggregates all the latency measurements as Jitter in both UL and DL and as the average round-trip time (RTT). In the UL Jitter (Fig. 5a) both SPGWU and P4-UPF had high spikes, in contrast with VPP which had more smooth performance. The low-jitter performance was also maintained in the DL jitter (Fig. 5b) and this also impacted the RTT (Fig.



(a) UL Jitter.



(b) DL Jitter.

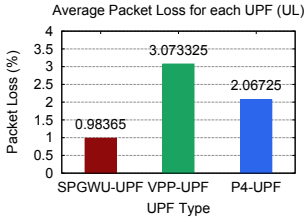


(c) Average RTT.

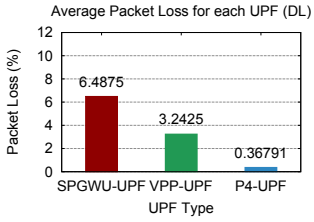
Fig. 5: Latency Evaluation: RTT & Jitter.

5c), with the VPP-UPF achieving the lowest average RTT value. Nonetheless, the jitter values of SPGWU and P4-switch UPF are not prohibitive.

Nevertheless, in terms of packet loss, the SPGWU had the lowest percentage in the UL traffic while the P4-switch UPF had the best performance in DL traffic, achieving only 0.3%. This is illustrated in Fig. 6a and 6b respectively. It is worth mentioning that all the metrics were observed at the same time. This means that the packet loss is mainly affected by the maximum throughput that we use to stress the links just before they break.



(a) Average UL Packet Loss.



(b) Average DL Packet Loss.

Fig. 6: Packet Loss Evaluation: Uplink and Downlink.

Finally, we measured the CPU consumption for the network functions only, since the P4-switch UPF offloads the packet processing to the chips. This means the chip itself is handling packet processing at line-rate speeds without involving the CPU. The CPU is typically only involved during the initial configuration or for control plane tasks. We specifically measured the CPU consumption for the SPGWU and VPP-UPF during UL UDP traffic. The results can be observed in Fig. 7. The VPP-UPF used the most CPU cores during the traffic reaching almost 170%. Furthermore, before and after the traffic, it occupied the CPU at 100% while being IDLE. This happens since the DPDK solutions such as VPP, often employ a polling mechanism rather than an interrupt-driven mechanism. This means VPP continuously checks for new packets to process, even if none are available. Polling can lead to higher CPU usage compared to interrupt-driven systems, especially when there's no traffic. In contrast with VPP-UPF, SPGWU had zero CPU consumption when IDLE in terms of traffic, and kept a relatively low CPU consumption percentage of 90% during traffic.

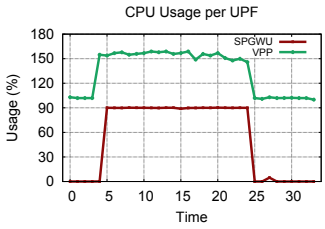
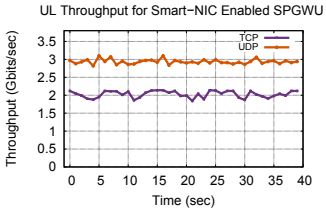


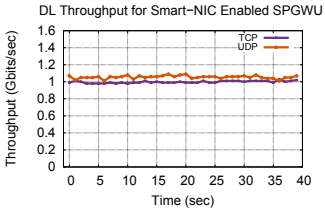
Fig. 7: CPU Consumption for each UPF.

B. SmartNIC P4-Card UPF (NITOS)

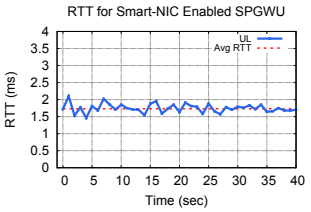
Testing the SmartNIC-P4 UPF gave us the most promising results in terms of Uplink throughput. As can depicted in Fig. 8a, the uplink throughput exceeded 3Gbps and 2Gbps in UDP and TCP respectively, outperforming any previous implementation. The downlink throughput (Fig. 8b) for both UDP and TCP is slightly lower than the one in the P4-Switch, but still one of the highest. In terms of RTT (Fig. 8c) the SmartNIC-P4 UPF showcased the highest value in comparison to the other implementations, while the jitter (Fig. 8d) seems to be similar and slightly improved in comparison to the one of the legacy SPGWU.



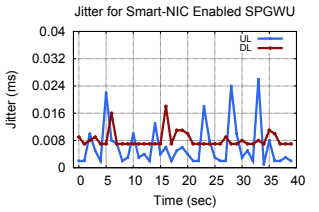
(a) UL Throughput for SmartNIC-P4 UPF



(b) DL Throughput for SmartNIC-P4 UPF



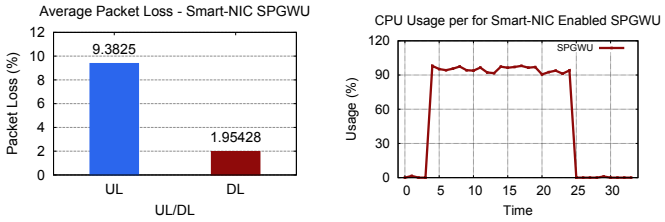
(c) RTT for SmartNIC-P4 UPF



(d) Jitter for SmartNIC-P4 UPF

Fig. 8: Throughput, RTT & Jitter for SmartNIC-P4 UPF

In the context of packet loss (Fig. 9a), it had the highest value in UL (exceeding 9%), whereas in DL, it had a relatively low percentage (1.9%). Finally, the CPU allocation (Fig.9b) was a bit higher than the one in legacy SPGWU-UPF, but lower than the one of the VPP-UPF.



(a) Average Packet Loss for SmartNIC-P4 UPF (b) CPU Consumption for SmartNIC-P4 UPF

Fig. 9: Packet Loss & CPU Consumption

Overall, we can conclude that the P4-switch UPF had an elite performance since it achieved high throughput, generally low latency values, and the lowest DL packet loss percentage. Additionally to these benefits, it doesn't occupy the CPU since it offloads all the data-plane packet processing to the chips. However, the VPP-UPF is a very promising solution since it's way cheaper than the P4-switch UPF, it has smooth performance on the latency, and also achieves high throughput rates, with the only drawback being the high CPU consumption. Finally, the SmartNIC-P4 UPF reached the highest throughput speeds in the UL scenario, and very good ones in the DL scenario (almost like the p4-switch). The jitter metrics showed a slight improvement in comparison with the default SPGWU, while the RTT and packet loss values were very high. This though, as the P4 implementation for the UPF, running on the 100Gbps switch, seems to be limited by its implementation. Similarly, performance is limited by the GPP cores available for traffic handling when using the VPP solution.

V. Conclusions

In this work, we studied and experimentally evaluated various UPF open-source solutions that rely on the legacy SPGWU, on the DPDK technology, and on the P4 language. Our experimental evaluation was under real-world 5G implementation using OpenAirInterface. Our findings revealed that the P4-based (P4-switch & SmartNIC-P4) UPFs generally showcased superior throughput and packet loss performance. This is a direct result of the hardware's ability to perform tasks faster and with less overhead, making it ideal for specialized functions that require rapid execution. However, the VPP-UPF presents a cost-effective alternative with competitive latency metrics. This study underscores the importance of selecting the appropriate UPF implementation while balancing performance, cost, and operational considerations. As 5G networks continue to evolve, further research and innovations in UPF implementations will be crucial in shaping the future of mobile communications. In the future, we foresee evaluating additional softwarized UPF solutions, with the most promising one being the eBPF/XDP-based UPF. In addition, our current findings may not fully reflect UPF performance on commercial platforms due to differences in network conditions and diverse cloud architectures. To provide a more thorough comprehension, future research will be held to offer a more detailed evaluation across varied deployment scenarios.

References

[1] N. Apostolakis, M. Gramaglia and P. Serrano, "Design and Validation of an Open Source Cloud Native Mobile Network," in IEEE Communications Magazine, vol. 60, no. 11, pp. 66-72, November 2022, doi: 10.1109/MCOM.003.2200195.

[2] H. T. Nguyen, T. Van Do and C. Rotter, "Scaling UPF Instances in 5G/6G Core With Deep Reinforcement Learning," in IEEE Access, vol. 9, pp. 165892-165906, 2021, doi: 10.1109/ACCESS.2021.3135315.

[3] T. Tsourdinis, N. Makris, S. Fdida and T. Korakis, "DRL-based Service Migration for MEC Cloud-Native 5G and beyond Networks," 2023 IEEE 9th International Conference on Network Softwarization (NetSoft), Madrid, Spain, 2023, pp. 62-70, doi: 10.1109/NetSoft57336.2023.10175417.

[4] Pupo, Irian & Santoyo-Gonzalez, Alejandro & Cervelló-Pastor, Cristina. (2019). A Framework for the Joint Placement of Edge Service Infrastructure and User Plane Functions for 5G. Sensors. 19. 10.3390/s19183975.

[5] Z. Wang and Y. Cai, "Management Optimization of Mobile Edge Computing (MEC) in 5G Networks," 2019 IEEE International Conference on Communications Workshops (ICC Workshops), Shanghai, China, 2019, pp. 1-6, doi: 10.1109/ICCW.2019.8756650.

[6] W. Zhu, P. Li, B. Luo, H. Xu and Y. Zhang, "Research and Implementation of High Performance Traffic Processing Based on Intel DPDK," 2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), Taipei, Taiwan, 2018, pp. 62-68, doi: 10.1109/PAAP.2018.00018.

[7] H. Zhang, Z. Chen and Y. Yuan, "High-Performance UPF Design Based on DPDK," 2021 IEEE 21st International Conference on Communication Technology (ICCT), Tianjin, China, 2021, pp. 349-354, doi: 10.1109/ICCT52962.2021.9657903.

[8] T. A. N. do Amaral, R. V. Rosa, D. F. C. Moura and C. E. Rothenberg, "An In-Kernel Solution Based on XDP for 5G UPF: Design, Prototype and Performance Evaluation," 2021 17th International Conference on Network and Service Management (CNSM), Izmir, Turkey, 2021, pp. 146-152, doi: 10.23919/CNSM52442.2021.9615553.

[9] D. Soldani et al., "eBPF: A New Approach to Cloud-Native Observability, Networking and Security for Current (5G) and Future Mobile Networks (6G and Beyond)," in IEEE Access, vol. 11, pp. 57174-57202, 2023, doi: 10.1109/ACCESS.2023.3281480.

[10] Toke Høiland-Jørgensen, Jesper Dangaard Brouer, Daniel Borkmann, John Fastabend, Tom Herbert, David Ahern, and David Miller. 2018. The eXpress data path: fast programmable packet processing in the operating system kernel. In Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '18). Association for Computing Machinery, New York, NY, USA, 54–66. https://doi.org/10.1145/3281411.3281443

[11] Jianer Zhou, Zengxie Ma, Weijian Tu, Xinyi Qiu, Jingpu Duan, Zhenyu Li, Qing Li, Xinyi Zhang, and Weichao Li. 2023. Cable: A framework for accelerating 5G UPF based on eBPF. Comput. Netw. 222, C (Feb 2023). https://doi.org/10.1016/j.comnet.2022.109535

[12] Robert MacDavid, Carmelo Cascone, Pingping Lin, Badhrinath Padmanabhan, Ajay ThakuR, Larry Peterson, Jennifer Rexford, and Oguz Sunay. 2021. A P4-based 5G User Plane Function. In Proceedings of the ACM SIGCOMM Symposium on SDN Research (SOSR) (SOSR '21). Association for Computing Machinery, New York, NY, USA, 162–168. https://doi.org/10.1145/3482898.3483358

[13] J. Rischke, C. Vielhaus, P. Sossalla, J. Wang and F. H. P. Fitzek, "Comparison of UPF acceleration technologies and their tail-latency for URLLC," 2022 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Phoenix, AZ, USA, 2022, pp. 19-25, doi: 10.1109/NFV-SDN56302.2022.9974798.

[14] UERANSIM: open source state-of-the-art 5G UE and RAN (gNodeB) simulator. The project can be used for testing 5G Core Network and studying 5G System. Available at: https://github.com/aligungr/UERANSIM

[15] TravelPing UPG-VPP: VPP-based User Plane Gateway. Available at: https://github.com/travelping/upg-vpp

[16] FD.io VPP: An Extensible Framework That Provides Out-of-the-Box Production Quality Switch/Router Functionality, FD.io, July 2017. Available: https://fd.io/docs/whitepapers/FDioVPPwhitepaperJuly2017.pdf

[17] S. Fdida, N. Makris, T. Korakis, R. Bruno, A. Passarella, P. Andreou, B. Belter, C. Crettaz, W. Dabbous, Y. Demchenko, and R. Knopp, *SLICES, a scientific instrument for the networking community*. *Computer Communications*, vol. 193, pp. 189-203, 2022.