Implementing Network Slicing in 5G Transport Networks Using Programmable Switches

Heitor Anglada, Raissa Costa, Cleverson Nahum, Silvia Lins and Glauco Gonçalves

***Abstract*— With the increasing complexity of 5G networks, flexible solutions are needed to ensure the requirements of each service. Network slicing isolates resources for different services, but there is no standard for end-to-end isolation in the transport network domain. This work explores using SDN and programmable P4 switches for network slicing in the transport domain. It presents a solution that accommodates various interconnections between the radio access and core network, evaluated in an experimental setup with Free5GC, UERANSIM, and Kathará. Results show the solution effectively maintains SLA requirements, enhancing flexibility in the 5G transport domain.**

***Keywords*— 5G Core, UPF, Network Slicing, P4, BMv2, Free5GC, Kathará, NSSAI.**

I. Introduction

The emergence of Fifth Generation of Mobile Networks (5G) did not represent a mere incremental evolution but a complete overhaul of network architecture, primarily focusing on digitalization, automation, and integration with other industries [1]. This new generation introduces advancements such as higher speeds, ultra-low latency (below 1 ms), massive support for connected devices (IoT), improved spectral efficiency, and new business opportunities. In this context, according to the ITU-R in [2], three primary use case categories stand out: Enhanced Mobile Broadband (eMBB), a direct evolution from 4G, offering higher transfer rates and greater bandwidth; Ultra-Reliable Low-Latency Communications (URLLC), designed for services requiring high reliability and extremely low latency; and Massive Machine-Type Communications (mMTC), covering applications involving a massive number of connected devices with extremely low power consumption. This increase in the variety of services supported by 5G networks demands a flexible, scalable, and modular infrastructure capable of simultaneously supporting diverse Quality of Service (QoS) requirements. To meet these demands, a modular 5G Core (5GC), Software-Defined Networking (SDN), and network slicing are adopted [3].

Unlike its predecessors, the 5GC adopts a Service-Based Architecture (SBA), in which each function offers and consumes services through standardized APIs. This allows for more dynamic and effective integration between functions, reinforcing modularity and facilitating network evolution as new demands arise [1]. To meet these demands, technologies such as network slicing have been implemented, allowing for logical and efficient network segmentation. Despite significant advances in 5G networking, slicing still faces challenges in the transport domain due to the lack of clear standards from organizations such as the O-RAN Alliance [4] and 3rd Generation Partnership Project (3GPP) [5], which currently define only the core and Radio Access Network (RAN) communication aspects, hindering effective end-to-end slice isolation.

While obtaining slicing information in the RAN and core domains is relatively straightforward, as Network Slice Selection Assistance Information (NSSAI) values are directly carried in packets, in the user plane, data is encapsulated in GPRS Tunneling Protocol (GTP) tunnels that do not explicitly contain the slice identifier. Traditional forwarding and routing strategies based on IP addresses or ports can be applied for basic traffic differentiation. However, such approaches may prove insufficient in specific scenarios, such as the one described in [5], where multiple distinct slices share the same User Plane Function (UPF) instance. In such cases, simple destination analysis does not ensure proper isolation, requiring additional mechanisms in the transport plane capable of inspecting deeper packet information to separate traffic efficiently.

This paper proposes using P4-programmable switches to implement network slicing in the 5G transport domain. The solution utilizes P4's capability to inspect the innermost headers of packets encapsulated in GTP User Plane (GTP-U) tunnels, dynamically segmenting traffic from different network slices based on the User Equipment's (UE's) source and destination IP addresses. The proposal's effectiveness is validated in an experimental test environment integrating Free5GC, UERANSIM, and P4 Behavioral Model (BMv2) switches. Finally, the results demonstrate proper isolation of slices representing eMBB and URLLC services while simultaneously meeting different SLA (Service Level Agreement) requirements guaranteed by each slice, even when they share a single UPF (User Plane Function) instance.

II. Related Works and Motivation

Recent research has explored the application of network slicing in 5G networks, examining various techniques and strategies to ensure data transport efficiency and isolation. Rommer *et al.* [1] detail the 5GC architecture and explore network slicing as a key enabler to meet diverse service requirements. Their discussion, however, is confined to the access and core domains, leaving unaddressed the challenges

Heitor Anglada, Raissa Costa, Cleverson Nahum and Glauco Gonçalves are with LASSE - Telecommunications, Automation and Electronics Research and Development Center, Belém-PA, Brazil . Silvia Lins is with the Innovation Center, Ericsson Telecomunicações S.A, Brazil. E-mails: heitor.anglada, raissa.cunha.costa@itec.ufpa.br, cleversonahum, glaucogoncalves@ufpa.br) silvia.lins@ericsson.com). This work was supported by the Innovation Center, Ericsson Telecomunicações S.A., OpenRAN Brazil - Phase 2 project (MCTI grant Nº A01245.014203/2021-14), SAMURAI project (FAPESP grant #20/05127-2), and CNPq-Brasil (grant 405111/2021-5).

associated with effective slicing within the transport domain. Wen and Yan [6] proposed an innovative architecture that implements the UPF directly on P4 switches, demonstrating significant performance gains over traditional software-based solutions. Nevertheless, their approach is limited to specific use cases and does not address the logical slicing of multiple services. A more recent work by Wen and Yan [7] improves upon this by addressing the logical segmentation of various slices, offering a more scalable and flexible solution for 5G network slicing.

NEC patented a transport-domain slicing solution presented in [8], which highlights using a single UPF; however, it adopts a different approach, separating flows based on the Tunnel Endpoint Identifier (TEID) and VLANs. The GSMA [9] states that there is currently no network slicing standard for the backhaul, although such standardization has existed for the core since the first 5G release and, more recently, in the RAN. Meanwhile, the O-RAN Alliance [4] proposes using VLANs or addressing slice isolation, given the lack of existing standardizations. On the other hand, the [5] only provides requirements for the transport domain, with the slicing standard being applied solely to the Core and RAN.

In contrast to related works that utilize VLANs or Tunnel Endpoint Identifiers (TEIDs), the main advantage of our approach lies in its ability to inspect the inner IP header of GTP-U encapsulated packets, allowing for precise forwarding that is independent of the UPF configuration. To the best of our knowledge, this is the first work to utilize the user's inner IP address as a routing key for network slicing directly in the transport plane, solving the challenge of multiple slices sharing a single UPF instance.

III. Transport Network Slicing using Programmable Switches

Since no established standard for slicing exists in the transport domain, solutions that implement isolation solely within this network segment are possible. Exploiting this gap, our work uses P4 switches to inspect and segment backhaul traffic, achieving end-to-end slice isolation independently of the core architecture and without relying on RAN components.

To differentiate the slices without modifying the packet, we adopted an approach that separates them based on the source and destination IP addresses. This approach allows the transport network to operate normally even when multiple slices share a single UPF instance. As illustrated in Figure 1, GTP packets do not carry any information about the slice to which they belong. Moreover, using P4 to segment the data plane based on the source IP range as a traffic differentiation criterion is not feasible with classical routing techniques, which only see the blue layer in Figure 1. In that figure, the orange fields—the source and destination IP addresses visible in the outer layers—are the only ones perceived by conventional switches and correspond to the gNB address.

We chose the P4 language to program these switches because it enables the extraction of data from the innermost layers of the packet, such as the inner IPV4 layer (red layer) in Figure 1. The packet headers had to be manually described,

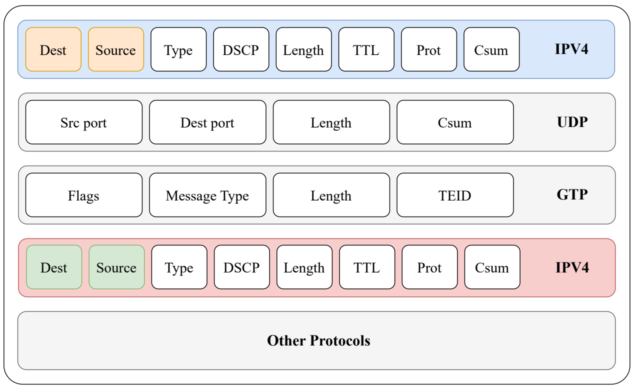

Fig. 1. Structure of a GTP packet that is routed in the transport domain, highlighting the fields used for slice separation with P4.

including all relevant protocol fields, to allow this inspection. With this setup, we used P4 to segment the transport plane and make forwarding decisions based on the green fields in the figure, representing the inner source IP address for uplink and the destination IP address for downstream, leveraging SDN switches to access this information and effectively separate the traffic.

Figure 1 helps visualize these layers. From top to bottom, the blue block represents the outer IPv4 header, which contains the protocol (prot), checksum (Csum) and the routing information used in the transport network; the UDP datagram includes a header and carries the GTP-U payload, where fields such as the TEID (the GTP tunnel identifier) are located. Deeper in the stack, the red block represents the inner IPv4 header, where the UE's source IP address appears, and contains the information that drives our slice-aware forwarding logic. Accordingly, in the P4 program, we declared each header according to the standard format: Ethernet with MAC addresses and EtherType; outer IPv4 (blue); UDP; GTP-U with TEID; and finally the `inner_ipv4` (red), which is used by the data plane tables to direct traffic for each slice.

This traffic separation using P4 is performed when the switch receives the packet and parses each protocol layer until it reaches the inner IPv4 header, highlighted in red. The source IP address is then extracted and compared with predefined ranges in the forwarding table. Based on this match, the switch determines the appropriate output port and forwards the packet through the corresponding path in the transport network until it reaches the UPF. On the return path, the logic is similar, but the forwarding decision is based on the inner destination (dest) IP address, ensuring that the packet follows the correct path from the UPF back to the gNB.

IV. Design and Deployment of the Transport Network Slicing

This section presents the implementation process of the proposed architecture to evaluate a network slicing strategy in the transport domain of 5G networks using BMv2 programmable switches with P4. Moreover, the BMv2 software can provide functional validation of P4 logic. Still, the collected performance metrics do not realistically reflects a dedicated P4 hardware. The main objective is to verify, through experiments monitoring the interfaces of network elements, the logical separation between slices based on the source IP address, dividing traffic in the data plane without replicating the UPF.

To achieve this, an experimental testbed was developed to simulate the main components of a 5G network, integrating tools such as Free5GC[1] for the network core, UERANSIM[2] for RAN and user emulation, and Kathará[3] to orchestrate the topology and BMv2 switches. The entire infrastructure was built using Docker containers, ensuring the portability and reproducibility of the experiments.

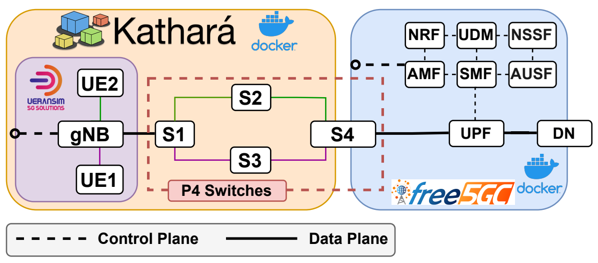

Fig. 2. Topology of the experimental environment showing the separation between the control and data planes.

As shown in Fig. 2, UERANSIM was used for the RAN simulation, implementing both the UE and the Next Generation NodeBs (gNB) in standalone mode. The simulator includes both control and user planes, logically separating communication between the UE and the core into the Access Stratum (AS), responsible for radio signaling and data traffic, and the Non-Access Stratum (NAS), which handles control messages between the UE and the Access and Mobility Management Function (AMF). UERANSIM supports key RAN interfaces, such as the user interface (for connecting to the UPF), the Service-Based Interface (SBI) interface (for service-based function communication), and the radio interface, which is simulated via the User Plane Protocol (UDP) protocol.

The network core was implemented using Free5GC, an open-source solution designed to comply with 3GPP specifications, enabling experimentation with 5G network scenarios without needing proprietary software or infrastructure. One of Free5GC's advantages is its ease of deployment in virtualized and containerized environments, using platforms such as Docker to emulate 5GC operations under different configurations. This characteristic facilitated testing in experimental networks and enabled the evaluation of service performance offered by the 5GC.

The gNB, UE, and the transport network were simulated using Kathará, a lightweight, container-based tool designed for rapid network topology creation. Kathará allows each network element to run as an isolated container, supporting pre-configured images such as P4 and UERANSIM, streamlining the configuration process.

The experimental setup comprises two UEs: one with SST = 1, associated with the eMBB network slice, and another

[1]`https://github.com/free5gc/free5gc`
[2]`https://github.com/aligungr/UERANSIM`
[3]`https://www.kathara.org/`

TABLE I
ALLOCATED THROUGHPUT PER EXPERIMENT FOR EACH NETWORK SLICE CONFIGURATION

Experiment	**Slice 1 (eMBB)**	**Slice 2 (URLLC)**
`Exp 1`	16 Mbps	4 Mbps
`Exp 2`	32 Mbps	8 Mbps
`Exp 3`	50 Mbps	10 Mbps
`Exp 4`	64 Mbps	16 Mbps

with SST = 2, with the URLLC network slice, as specified in Table II. The table lists the Tracking Area Code (TAC), the Mobile Network Code (MNC), and each UE identifier. Traffic generation is performed using `iperf` with configured throughput values ranging from 16 to 64 Mbps per slice, as defined in Table I, while maintaining logical separation in the transport network. However, since both slices terminate at a shared UPF instance, conventional destination-based IP routing proves insufficient for traffic differentiation. Consequently, packet forwarding decisions are made by extracting source IP addresses from encapsulated GTP-U headers. This approach necessitates BMv2 programmable switches with deep packet inspection capabilities to process inner packet headers.

In this scenario, two Docker networks were used: one dedicated to Kathará and another to Free5GC. Kathará manages the containers for the UEs, gNB, and P4 switches, while Free5GC hosts the core functions. Switch S4 interconnects the two networks, as shown in Fig. 2, through three interfaces: eth0 and eth1 connected to Kathará (each associated with a slice), and eth2 connected to the UPF in Free5GC.

The transport network consists of four switches: S1 receives traffic from the gNB and, based on P4 logic, classifies flows according to the encapsulated source IP address. Depending on the slice, traffic is forwarded to either switch S2 (eMBB) or S3 (URLLC), which then directs it to switch S4 for delivery to the UPF. The gNB performs GTP encapsulation and communicates with the AMF and the UPF, using separate interfaces (eth1 and eth0, respectively) to ensure isolation between the control and user planes.

This architecture enables scalable and flexible slicing in the transport plane using a single UPF instance. GTP-U headers are parsed in the P4 pipeline to extract the UEs' actual source IP addresses, allowing slice-aware forwarding without modifying the core network or replicating UPF instances. This approach ensures that the QoS requirements of different slices can be met through programmable logic in the data plane, validating the efficiency and feasibility of the proposed solution. Tests were conducted to evaluate the implementation and measure the CPU usage of the P4 switches and the performance of each slice in terms of latency and throughput during concurrent operation, as detailed in the following section.

V. Results and Discussion

For the tests, traffic was generated by an `iperf` client running on each UE, with the corresponding servers hosted in the UPF container. The packet was captured using `tcpdump` at both the UE egress and the UPF ingress, allowing observation of traffic behavior along the data path. This capture

TABLE II
USER EQUIPMENT (UE) CONFIGURATION PARAMETERS

Field	Value	Description
SUPI	208930000000001	Subscription Permanent Identifier (MCC + MNC + IMSI)
MCC	208	Mobile Country Code (Identifies the country)
MNC	93	Mobile Network Code (Identifies the mobile operator)
IMSI	0000000001	International Mobile Subscriber Identity (Unique subscriber identifier)
TAC	1	Tracking Area Code (Geographic location code within a PLMN)
SST (UE1)	1	Slice Service Type: Enhanced Mobile Broadband (eMBB)
SST (UE2)	2	Slice Service Type: Ultra-Reliable Low Latency Communications (URLLC)
SD (UE1)	010203	Slice Differentiator for eMBB (Unique identifier within the slice type)
SD (UE2)	112233	Slice Differentiator for URLLC (Unique identifier within the slice type)
DNN	internet	Data Network Name (Name of the external data network)
IP Range (UE 1)	10.60.0.0/16	Allocated IP address range for UE 1
IP Range (UE 2)	10.61.0.0/16	Allocated IP address range for UE 2

methodology provides the granular data to validate how P4-programmable switches could optimize traffic flows. The resulting capture files were processed by a Python script that extracted throughput and latency information for every packet.

To enable an accurate analysis, the data from each of the four experiments described in Table I were grouped into 100 ms windows. The average latency and throughput values were computed for each interval, yielding more representative plots and making it easier to visualize the time-varying behavior.

Three key metrics were considered to validate the proposed architecture and assess its computational impact relative to switches performing simple packet forwarding: slice throughput, slice latency, and switch CPU usage. These metrics are particularly relevant for demonstrating P4's advantages in dynamic traffic separation. Four experiments were carried out in which the transfer rates applied to each slice were progressively increased, as shown in Table I. It is important to highlight that the switch handling eMBB slice traffic (S2) is configured for a maximum rate of 50 Mbps and a latency of up to 10 ms. In comparison, the switch forwarding URLLC slice traffic (S3) is set to support a maximum rate of 10 Mbps with 1 ms latency.

Figure 3 shows the traffic curves for these two slices in Experiments 2 (yellow curves) and 4 (green curves), as these cases also represent the behavior observed in Experiments 1 and 3. For the eMBB slice curves (marked with an X), the green curve shows the transfer rate staying close to the 32 Mbps generated by `iperf`, while the yellow curve, with 64 Mbps of offered traffic, clearly respects the 50 Mbps cap configured in switch S2. Similarly, the curves marked with a dot illustrate the behavior of the URLLC slice: the rate follows the offered traffic in the green curve (8 Mbps), but in the yellow curve, when 16 Mbps is injected, it is limited to 10 Mbps, as defined in switch S3.

When analyzing the latency data in Table III, we observe that the eMBB slice behavior varies according to the applied transmission rate. In Experiment 2 (32 Mbps), the average latency remains stable at 11.65 ms with a low standard deviation of 1.52 ms, indicating the infrastructure operates within the switches' processing capacity. In contrast, Experiment 4 (64 Mbps) shows the average latency increasing to 132.76 ms with a 10.42 ms standard deviation - clear evidence of transport plane congestion, precisely the scenario where P4-enabled slicing prevents cross-slice interference.

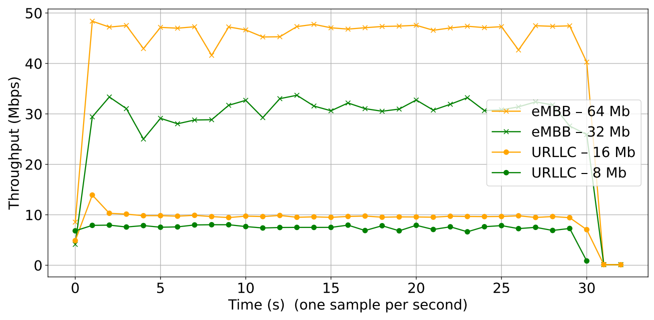

Fig. 3. Traffic observed on the eMBB and URLLC slices during Experiments 3 and 4.

TABLE III
SUMMARY STATISTICS (MEAN AND STANDARD DEVIATION) OF LATENCY AND THROUGHPUT PER SLICE AND EXPERIMENT

	Mean		StdDev	
	Slice 1	**Slice 2**	**Slice 1**	**Slice 2**
`Exp 1`	10.73 ms	2.66 ms	0.32 ms	0.10 ms
`Exp 2`	11.65 ms	2.73 ms	1.52 ms	0.36 ms
`Exp 3`	132.05 ms	758.03 ms	12.59 ms	229.11 ms
`Exp 4`	132.76 ms	853.68 ms	10.42 ms	152.73 ms

Table III presents results for the URLLC slice, which has significantly stricter latency requirements. At 4 Mbps and 8 Mbps rates, the average latency remains at 2.66 ms and 2.73 ms, respectively, with low standard deviations (0.10 ms and 0.36 ms), demonstrating stability and compliance with established Service Level Agreementss (SLAs). However, in the 10 Mbps and 16 Mbps experiments, latency increases dramatically to 758.03 ms and 853.68 ms, respectively, with high standard deviations (229.11 ms and 152.73 ms). These results highlight the critical role of P4-programmable switches in maintaining slice isolation under excessive load conditions. They should be interpreted as a validation of the isolation between slices, and not as absolute performance benchmarks that would be achievable on hardware.

Finally, several noteworthy points are revealed by the CPU-usage plot for the containerized switches (Figure 4). First, although switches S1 and S4 handle the same traffic volume during the 0–15 s interval, their resource usage differs slightly. This overhead is expected, as S1 performs the more complex task of deep packet inspection, analyzing multiple headers (Ethernet, outer IPv4, UDP, GTP) to access the inner IP, whereas S4 merely relays packets arriving from S2 and S3 to the core network. The additional processing complexity at S1 translates into roughly a 10% higher CPU load than S4.

The figure also shows that except for S1—which processes all incoming traffic—the CPU consumption of the other switches converges to a plateau. This behavior is tied to the throughput cap each slice can sustain. Once the first bottleneck in the data path is reached, the overall system throughput is limited, and the resource usage of downstream elements levels off accordingly.

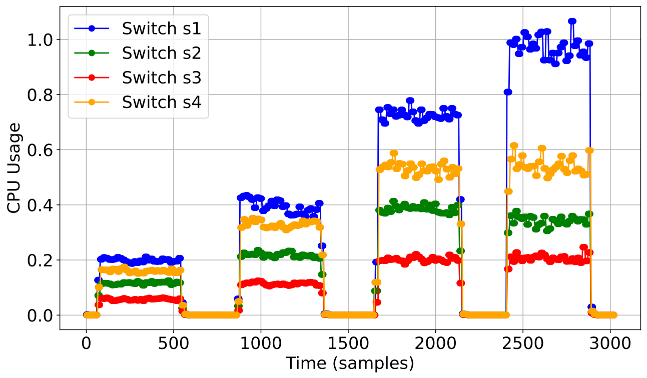

Fig. 4. CPU usage of switch containers during experiments, showing increased load on Switch S1 due to packet inspection and slice-aware forwarding with P4.

Although the experiments were conducted with two slices, the forwarding logic based on the inner IP ranges can be easily scaled. The practical bottleneck would be the flow table capacity and the switch's processing power, not the separation logic itself. In topologies with multiple UPFs, the approach remains valid, simply by configuring the rules to direct each slice to the corresponding UPF. Managing dynamic scenarios, such as handover and user mobility, would require integration with a dynamic control plane (such as SDN networks) to update the switches' rules in real-time.

VI. Conclusions

This work addressed the challenge of effectively implementing network slicing in the 5G transport domain. In this scenario, the absence of clear standards and the encapsulation of user plane traffic in GTP tunnels without explicit slice identifiers hinder end-to-end isolation, especially when multiple slices share the same UPF instance. To overcome these limitations, we propose using P4-programmable switches capable of performing deep packet inspection and dynamically segmenting traffic from different network slices based on the UE's internal IP addresses directly in the transport data plane.

Indeed, the experimental validation in an environment combining Free5GC, UERANSIM, Kathará, and BMv2 switches demonstrated the feasibility of slice-aware forwarding in 5G transport, even with a single UPF. The results indicate that, by parsing GTP-U headers and using the inner IP address as a forwarding key, the approach successfully segregated the traffic of eMBB and URLLC slices via independent paths. Performance was maintained within the configured limits, with average latencies of approximately 11 ms for eMBB and 2 ms for URLLC under moderate load. Additionally, CPU usage on the forwarding switches (S2, S3, S4) leveled off due to their rate caps, while the P4 logic added approximately 10% processing overhead on the inspection switch (S1) compared to an unprogrammed counterpart. Future work will explore scenarios with more slices and shared UPFs, as well as integrating dynamic orchestration APIs so that paths and policies can automatically adapt to changing traffic demands.

References

[1] S. Rommer, P. Olsson, P. Frenger, J. Sachs, G. Rune, and M. Skold, *5G Core Networks: Powering Digitalization*, 2nd ed. Academic Press, 2020.

[2] "Imt vision—framework and overall objectives of the future development of IMT for 2020 and beyond," International Telecommunication Union, Radiocommunication Sector (ITU-R), Recommendation ITU-R M.2083-0, 2015, accessed: 05 May. 2025. [Online]. Available: https://www.itu.int/rec/R-REC-M.2083-0-201509-I/en

[3] H. Zhang, N. Liu, X. Chu, K. Long, A.-H. Aghvami, and V. C. M. Leung, "Network slicing based 5g and future mobile networks: Mobility, resource management, and challenges," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 138–145, 2017.

[4] O-RAN Alliance, "O-RAN WG1 Slicing Architecture Specification," O-RAN Alliance, Tech. Rep. O-RAN.WG1.TS-Slicing-Architecture-R004-v14.00, 2023, accessed: 2024-04-01. [Online]. Available: https://specifications.o-ran.org/

[5] 3GPP, "TS 28.530 V19.0.0: Management and Orchestration; Concepts, Use Cases and Requirements," 3rd Generation Partnership Project (3GPP), Tech. Rep., March 2025. [Online]. Available: https://www.3gpp.org/DynaReport/28530.htm

[6] R. MacDavid, C. Cascone, P. Lin, B. Padmanabhan, A. ThakuR, L. Peterson, J. Rexford, and O. Sunay, "A p4-based 5g user plane function," in *Proceedings of the ACM SIGCOMM Symposium on SDN Research (SOSR)*, ser. SOSR '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 162–168. [Online]. Available: https://doi.org/10.1145/3482898.3483358

[7] Z. Wen and G. Yan, "HiP4-UPF: Towards High-Performance comprehensive 5g user plane function on p4 programmable switches," in *2024 USENIX Annual Technical Conference (USENIX ATC 24)*. Santa Clara, CA: USENIX Association, Jul. 2024, pp. 303–320. [Online]. Available: https://www.usenix.org/conference/atc24/presentation/wen

[8] S. Ishikura, A. Shiroyama, and N. Naohiro, "User data processing device, network interface, and method," May 18 2023, uS Patent App. 17/917,291.

[9] GSMA, "NG.135 Version 3.0: E2E Network Slicing Requirements," GSM Association (GSMA), Tech. Rep., June 2023. [Online]. Available: https://www.gsma.com/newsroom/wp-content/uploads//NG.135-v3.0-3.pdf