SliceNetVSwitch

Matencio Escolar, Antonio; Wang, Qi; Alcaraz Calero, Jose M.

Published in:
IEEE Transactions on Network and Service Management

DOI:
10.1109/TNSM.2020.3029653

Published: 09/12/2020

Document Version
Peer reviewed version

Matencio Escolar, A., Wang, Q., & Alcaraz Calero, J. M. (2020). SliceNetVSwitch: definition, design and implementation of 5G multi-tenant network slicing in software data paths. *IEEE Transactions on Network and Service Management*. https://doi.org/10.1109/TNSM.2020.3029653

SliceNetVSwitch: Definition, Design and Implementation of 5G Multi-tenant Network Slicing in Software Data Paths

Antonio Matencio Escolar, Qi Wang, Jose M Alcaraz Calero, *Senior Member, IEEE*

***Abstract*—Network slicing is a primary Fifth-Generation (5G) mobile networking technology to create virtualised and softwarised logical networks for various vertical businesses with diverging Quality of Service (QoS) requirements. Meanwhile, there is a clear gap in providing network slicing capabilities in 5G multi-tenant networks to enable guaranteed QoS in terms of well-defined network metrics for the multiple tenants sharing the same physical infrastructure. This paper designs and implements novel software data path architecture that enables such network slicing with assured QoS in 5G multi-tenant networks. Highly flexible and customisable definition of network slicing is also allowed to be aligned with different existing definitions on demand and at run time. The proposed architecture has been prototyped based on the popular Open Virtual Switch (OVS), and empirically validated to demonstrate the deployment and management of network slices with the above capabilities. Intensive scalability results are provided where more than 8,192 network slices are achieved simultaneously with warranted QoS through performance isolation in terms of bandwidth and delay in a real softwarised 5G multi-tenant infrastructure at speeds of up to 10 Gbps.**

***Index Terms*—Network Slicing, 5G, Software-Defined Networks (SDN), Network Function Virtualization (NFV), OpenVSwitch (OVS), OpenFlow, Software data path, Quality of Service (QoS).**

I. Introduction

In the upcoming years, the emerging Fifth-Generation (5G) technology will change the ecosystem of mobile networks, transforming completely the current business models. 5G technology is expected to provide a wide range of new tailored services to a wide range of vertical businesses such as Industry 4.0, Autonomous Driving, Smart Cities, Smart Grid, IoT Technology and Tele-surgery to mention a few. In this context, a vertical is a business that serves a specific set of final customers and demands a specific set of requirements to make it happens. These new use cases have extremely demanding Quality of Service (QoS) requirements in terms of latency, bandwidth, reliability, scalability and flexibility. To meet these ambitious requirements, ITU (International Telecommunication Union) has classified three main use case categories: eMBB (Enhanced Mobile Broadband), mMTC (Massive Machine Type Communications) and URLLC (Ultra-Reliable Low-Latency Communications) [1]. In a URRLC 5G ambulance use case scenario, a paramedic is wearing a headset with camera to deliver video to the doctors at the hospital while it is in the ambulance with the patient to allow early remote tele-assessment while mobile on the road until arriving to the hospital. The network should be able to provide a reliable and timely service for patient monitoring, audio and video. This imposes high requirements in terms of low latency, reliability and warrantied bandwidth and should also ensure mobility across gNBs and different mobile operators. We will use this use case along this paper to illustrate how our approach fits in the context of E2E network slicing.

It is widely recognised that traditional networks are based on the "one-size-fits-all" paradigm and thus not flexible enough to address the myriad of new use cases that are classified under the umbrella of these three ITU categories [2]. This problem is exacerbated by the fact that 5G architecture is based on softwarisation and virtualisation, using Software-Defined Networks (SDN) and Network Function Virtualisation (NFV) ([3], [4]), as a cornerstone paradigm to reduce capital expenditures (CAPEX). Softwarisation and virtualisation have enabled the capabilities to share physical infrastructures by multiple tenants (e.g., network operators) in a multi-tenant environment to further reduce both CAPEX and operating expenditure (OPEX) through virtualisation/containerization [5].

The shift to the novel 5G network slicing paradigm will enable providing customised and warrantied QoS "as-a-service", in terms of well-defined QoS metrics such as bandwidth, latency and jitter, for 5G mobile operators and their customers, especially vertical businesses. To achieve this game-changing network slicing paradigm, several enabling technologies are required along the whole data path of the 5G network. This paper proposes an enabler for the software data path that interconnects virtual and physical machines to provide QoS warranties among operators sharing a physical machine and between vertical sectors using the services provided by such operators.

Currently, the most advanced software data paths allow warranting both bandwidth and latency in traditional IP networks by intelligently using packet classifiers and queuing disciplines [6] where the rules are applied to the same IP traffic along the different network segments.

However, 5G multi-tenant networks imply the use of overlay networks to deal with both 5G user mobility and tenant isolation. Overlay networks are networks inside of networks and thus, vertical in their nature. In multi-tenant infrastructures, each tenant has a dedicated overlay virtual network for their own purposes whereas all overlay networks share the same

Antonio Matencio Escolar, Qi Wang and Jose M. Alcaraz Calero are with School of Computing, Engineering and Physical Sciences, University of the West of Scotland, PA1 1LU, Paisley, United Kingdom.

E-mail: {antonio.matencio, qi.wang, jose.alcaraz-calero}@uws.ac.uk

underlying physical network. Analogously, in 5G networks, each 5G user equipment (UE) has a dedicated logical connection to the User Plane Function (UPF) to allow mobility across antennas and thus, maintain connectivity whereas all the 5G traffic is shared over the same physical network. In these contexts, the definition of a network slice represents the association of overlay traffic with QoS warranties e.g., in terms of bandwidth, latency and jitter.

The main problem addressed in this paper is that there is no existing software data path that allows a flexible definition of network slices in a 5G multi-tenant infrastructure. The direct implications of this lack of support is multi-fold: no fine-grained control of 5G traffic or tenant traffic is possible, neither 5G users nor tenants have performance protection capabilities, and consequently, there is no support for the new use cases envisioned for 5G. These are the main motivations of this paper. Accordingly, the main contribution of this work is the design and implementation of a novel software data path that brings a significant number of innovations beyond the current state-of-the-art, summarised as follows:

- Providing a flexible and programmable definition of network slices in 5G multi-tenant networks by enhancing traditional packet classifiers to support 5G overlay networks.
- Extending the current programmability of the network control to support enforcing QoS (bandwidth, latency and jitter) warranties in 5G overlay networks.
- Realising a management interface where the life-cycle of network slices can be controlled in real time.
- Prototyping the above innovations in a well-known software data path, OpenVSwitch (OVS) as a proof of concept.
- Validating and demonstrating the proposed solution in a 5G multi-tenant testbed where both functional testing and scalability results have been analysed.

The remaining of this paper is organized as follows: Section II reviews the different approaches to the concept of network slicing in the current state of the art and introduces our novel proposed definition of network slice. Section III explains the 5G multi-tenant architecture proposed in this paper, the different segments it consists of and the scope of the software datapath within this architecture. This section also includes an overview of the current state-of-the-art regarding software data paths and a comparison to our proposed solution. Then, section IV describes the proposed 5G software data path architecture. Section V provides implementation details to achieve network slicing and explains the deployed scenarios to validate our contribution and the empirical results achieved in terms of functional and scalability validation. Finally, section VI concludes the paper.

II. Network Slicing: Literature review and proposed definition of network slice

A. Existing Definitions of Network Slicing

An increasing number of studies such as Rost et al. [7], S. Zhang [8] and Afolabi et al. [4] have indicated the concept of network slice as one of the most important pillars in 5G networks. However, there is no unified vision or consensus, and in consequence, not yet a clear and precise definition of network slices, leading to various perspectives.

For instance, according to NGMN Alliance [9] (Next Generation Mobile Networks), a network slice is defined as a set of network functions running on top of physical resources where both, network functions and resources form a logical network to meet some network characteristics. GSMA [10] complements the definition of network slicing from a mobile operator's point of view, as an independent end-to-end logical network that runs on a shared physical infrastructure, capable of providing a negotiated service quality, which indicates that a network slice comprises dedicated and/or shared resources, e.g. in terms of processing power, storage, and bandwidth and has isolation from the other network slices. IETF [11] defines a network slice as a managed group of subsets of resources, network functions at the data, control, management/orchestration and service planes at any given time. In the paper entitled "5G-TRANSFORMER: Slicing and Orchestrating Transport Networks for Industry Verticals" [12], the authors propose a complementary concept of network slice from the perspective of a vertical industry as *"a dedicated logical infrastructure provided to verticals to support their services and meeting their specific requirements"*.

Therefore, the definition of a network slice can vary depending on the business roles. For vertical businesses, a network slice can be perceived as a customised service to fulfill their performance requirements in terms of QoS such as bandwidth and latency, coverage and network functions such as encryption, load-balancing, caching and so on. For Infrastructure Service Providers, a network slice could be defined as the services dedicated to and consumed by a particular tenant that needs to fulfill a set of specific requirements. Analogously, for Mobile Network Operators (MNOs), a network slice could be the services dedicated to/consumed by specific User Equipment (UE) that needs to fulfill the user's requirements.

It is clear that a 5G infrastructure is a heterogeneous ecosystem where many different business roles and technologies, both hardware and software, coexist. Network slicing should be defined either implicitly or explicitly as an end-to-end concept that must be implemented in all the different segments of the network infrastructure including Radio Access Network (RAN), Fronthaul, Edge, Transport, Core and Backbone networks.

B. Proposed Definition of the Scope of a Network Slice

Our contribution complements the definitions of network slicing in subsection II-A by providing a more accurate and precise, yet flexible definition of the scope, in terms of network traffic over the data path, that conforms a given network slice. Thus, our definition is as follows: *"a network slice in the data plane is any network flow or aggregation of flows that represents the traffic of a specific user or a specific user's service or groups of users' services, a specific tenant or a specific tenant's service, specific infrastructure or infrastructure's service that can be uniquely identified, isolated and controlled to warranty a set of service level agreements in*

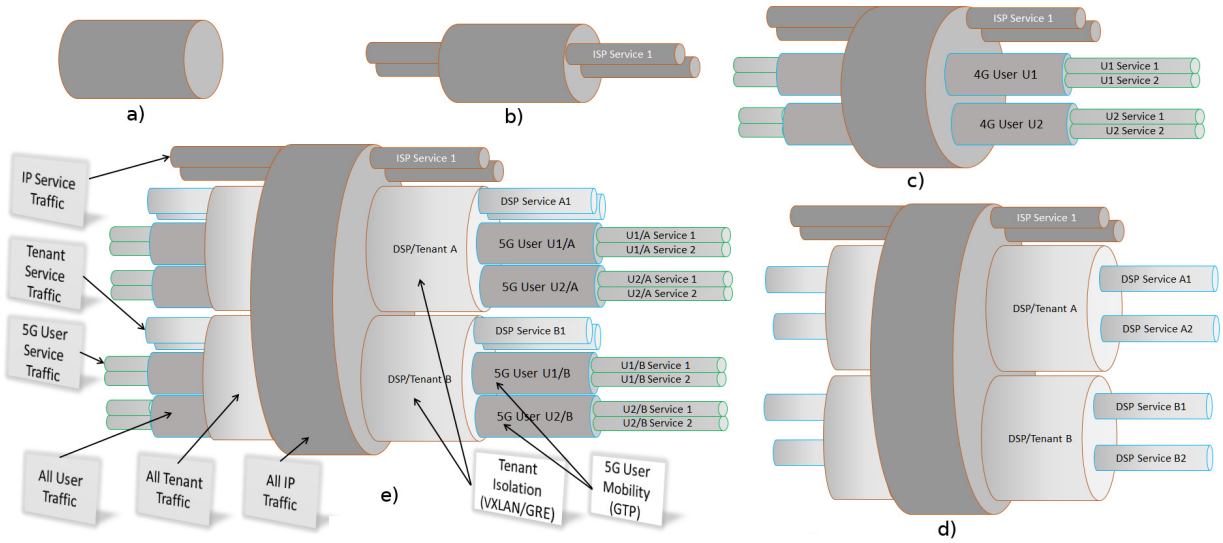

Fig. 1. Scenarios of different infrastructure data paths: a) traditional best-effort IP network; b) service differentiation IP network; c) traditional LTE/4G network with QoS; d) multi-tenant infrastructure with QoS support; e) 5G multi-tenant infrastructure with QoS support

any of the network segments of the infrastructure, regardless of the technology, packet structure or encapsulation protocols employed". The proposed definition, firstly, includes and takes into account all previous definitions found in the current state of the art and, secondly, is more accurate and precise that previous definitions due to the flexibility enough to deal with all types of traffic that can be identified in a 5G architecture in order to cope with verticals use cases and requirements.

Figure 1 is composed by five different scenarios to show graphically how the different data paths evolve in complexity. Scenario a) is a *traditional best-effort IP network* where all the traffic is equally controlled in a "one-size-fits-all" paradigm. Scenario b) corresponds to the *service differentiation IP network*, usually implemented by the IP DiffServ header where services are differentiated and controlled according to QoS policies. Scenario c) represents the data path of a *traditional LTE/4G network with QoS support* provided by the Policy and Charging Rules Function (PCRF). The PCRF allows assigning QoS traffic rules to different users and their services in the 4G networks. This scenario fits the case where a mobile operator manages its own infrastructure without sharing resources with any tenant. Scenario d) represents the data path of a *multi-tenant infrastructure with QoS support* where different digital service providers (DSP) share the same infrastructure offered by an infrastructure service provider (ISP). Both DSP and ISP services can be treated differently according to QoS policies. Finally, Scenario e) represents the data path of a 5G multi-tenant infrastructure with QoS simultaneous support at multiple levels of granularity, e.g. at physical infrastructure level, at multi-tenant level, at 5G users level and at 5G users belonging to each of the virtual mobile network operators to meet the use case requirements demanded by different vertical businesses and service providers. Our definition of the scope of a network slice is flexible enough to cover all these scenarios and to the best of our knowledge, it is the first time to provide a framework with support for these capabilities.

In terms of innovation, scenario b) is widely referred to as Quality of Service/Differentiated Service Code Point (QoS/DSCP) control over traditional networks although some studies are starting to refer to it as network slicing. Our definition is a significant extension to this view by allowing different business roles to define the scope of network slices respectively in all the scenarios described. Scenario b) is considered as the state of the art, scenario c) as a very advanced deployment of 4G networks, and Scenario d) as experimental features[1]. Only scenario e) is flexible enough to be able to provide user mobility, multi-tenant isolation and fine grain control over 5G traffic and thus, offer an E2E network slicing service and it is usually associated to virtualised mobile network operators. However, to the best of our knowledge, this paper is the first one to propose and achieve scenario e). Our proposed architecture provides support for all these scenarios. In fact, to fill this significant gap in the support for all of them has been our main motivation.

III. Software Data Paths: Overview of a 5G Multi-tenant Software Data Path and related work on Software Data Paths

A. Dissection of a 5G architecture: network segments and architectural components

Figure 2 shows an overview of the different network segments of a 5G multi-tenant network, focused on the data plane for the 5G users. From the left to the right, 5G UE devices are connected to Distributed Units (DUs) via the 5G New Radio interface. In our use case, they are headset devices hold by the paramedics in the 5G ambulance to deliver video to the hospital doctor in real-time allowing early remote tele-assessment while in the road. DUs provide connectivity to the edge network segment. The edge is considered the last

[1]OpenStack Network and further release provide partial support for this capabilities https://docs.openstack.org/mitaka/networking-guide/config-qos.html

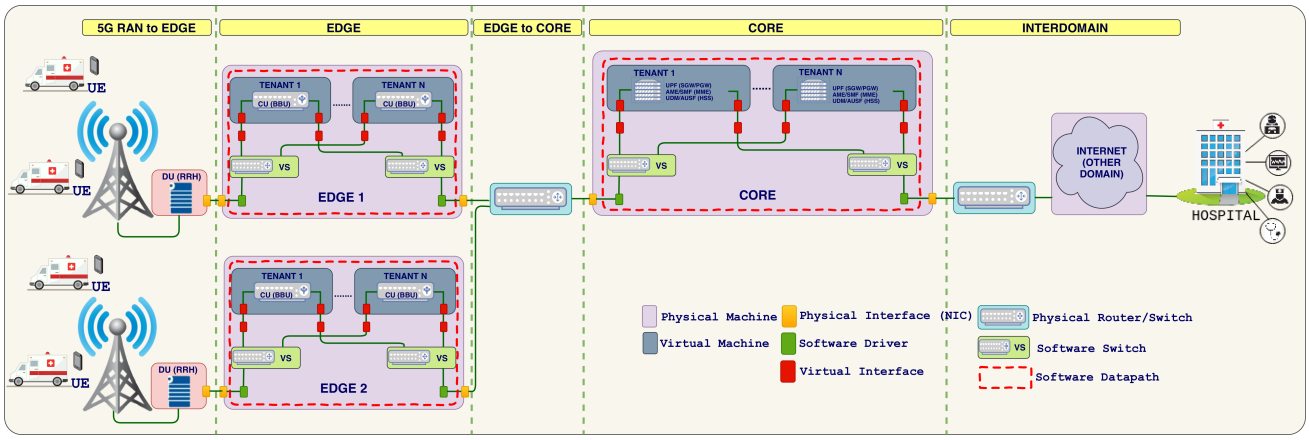

Fig. 2. Overview of 5G multi-tenant data path along 5G network segments

mile, close to the final users where Commercial Off-The-Shelf (COTS) computers can allocate different components of the 5G architecture. The 5G architecture is characterised by two distinct features. Firstly, architectural components have been softwarised and now can run on top of software using commodity containers/virtual machines. Secondly, COTS computers are now shared in a multi-tenancy environment where multiple telecommunication operators share the same physical infrastructure through virtualisation and tenancy isolation. The Multi-access Edge Computing (MEC) architecture, directly associated to 5G multi-tenant networks, extends these two features to the edges of the network. Thus, the different physical machines in Figure 2 allocate different softwarised Centralized Units (CUs), associated to different tenants. CUs deal with the management of the radio interface, user handovers and security, among others, usually refereed as gNB. To this end, CUs create an overlay network to handle the mobility of the user across DUs. The edge is connected to the core, usually by a transport network. The core network is a traditional cloud computing infrastructure. In Figure 2, each 5G core has a set of User Plane Function (UPF) in charge of forwarding user data in the data path and a set of control and management functions in charge of authentication, user mobility, handover management, user registry, and so on [13]. In the data path, UPF acts as a mobility anchor for the 5G users and as a gateway where they get access to the Internet. The figure shows the associated naming of the 5G functionality to the 4G equivalent in parenthesis. Thus, to warranty the quality of the video transmission across the whole network, in every single network segment, it is required a complete end-to-end network slicing solution involving: slicing of the radio interface from the paramedic device located at the ambulance, passing by the slicing in the edge network, in the transport network, in the core-network and in the multi-domain network until the doctor located at the hospital. See Figure 2 with all the network segments. The coordination of the control functions in charge of the enforcement of network slicing across different network segments to achieve a true E2E network slicing is carried out by the management and orchestration (MANO) architecture. In this context of E2E network slicing, our contribution is focused on the control functions in charge of the enforcement of network slicing in both edge and core network segments.

B. Virtualisation and softwarisation: key role and challenges in 5G networks

Softwarisation and virtualisation has imposed the usage of software data paths to interconnect the virtual interfaces of virtual machines (VMs) and containers to the interfaces available in the physical machines. This is currently happening in both edge and core network segments. Figure 2 shows the area covered by a particular software data path in both edge and core network segments, plotted with dotted lines. Two software switches are available in each of the physical machines, and they are used to interconnect the physical network interface card (NIC) with the different virtual interfaces exposed by the hypervisor to allow VM traffic to go in/out by mean of a software switch that provides the same functionalities as an Ethernet switch. The figure also shows how different VMs, belonging to different tenants, are connected to the same software switches. This imposes significant challenges. Firstly, traffic isolation needs to be enforced for security purposes using overlay networks. Secondly, filtering and charging policies need to be applied for security and auditing purposes. Thirdly, performance isolation should be carried out to ensure that VMs make the best controlled and optimised usage of the available physical resources. The first two features can be considered the state of the art. However, the third one is not yet available is the current cloud computing stacks. The situation becomes more complex when 5G infrastructures are deployed therein, and then UPF and DU components (shown in Figure 2) create an overlay network to isolate the network traffic of all the mobile users among them. 5G networks need to provide network slicing by design and currently there is no such fine-grained 5G user level control of network slicing in softwarised data paths. This is the main contribution of this work.

C. Related Work on Software Data Paths

Table I provides a detailed comparative analysis of the main data path implementations available against the key capabilities required to provide support for network slicing. Such

TABLE I
SOFTWARE DATAPATH CAPABILITIES FOR 5G NETWORK SLICING

	Classification capabilities											Control and actions capabilities							Other Features	
Datapath	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	API	Execution Environment
Netmap [14]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	C Syscall	Kernel
PF Ring [15]	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	C	Kernel
DPDK [16]	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗	✗	✓	✗	C API	User
OpenVSWitch	✓	✗	✓	✓	✗	✗	✗	✓	✗	✗	✗	✓	✓	✓	✓	✗	✓	✓	OpenFlow	Kernel & User
eXpress DataPath (XDP) [17]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✗	✓	✗	eBPF	Kernel
Linux Traffic control (TC) [18]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	eBPF & U32	Kernel
Linux Netfilter (IP Tables) [19]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✗	✓	✗	eBPF & U32	Kernel
Our Contribution	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	OpenFlow	Kernel & User

TABLE II
5G NETWORK SLICING CAPABILITIES

	Classification Capabilities	**Matching Headers**	**Type of Traffic Classified**
1	Traditional IP Traffic	Ether/IP[/UDP][/TCP]	Traditional user and/or services
2	Traditional DiffServ (DSCP)	IP DSCP	QoS in Traditional services
3	5G User DiffServ (DSCP)	5G User DSCP	QoS in 4G/5G User's services
4	5G User Mobility Network (UMN)	GTP	5G Users
5	IP traffic in UMN	5G User IP[/UDP][/TCP]	5G User's services
6	5G User Mobility Overlay Network (UMN) over a Multi-Tenant Overlay Network (MON)	GTP over VXLAN/GRE	5G User connected to a Tenant network
7	IP traffic in UMN over MON	IP[/UDP][/TCP] of a 5G User of a Tenant	5G User's services connected to a Tenant network
8	Multi-Tenant Overlay Network (MON)	VXLAN/GRE	Tenants
9	IP traffic in MON	Tenant Ether/IP[/UDP][/TCP]	Tenant's users and/or services
10	Tenant DiffServ (DSCP)	Tenant DSCP	QoS in Tenant's services
11	5G User DiffServ (DSCP) over a Tenant	5G User DSCP over VXLAN/GRE	QoS in 5G User's services connected to a Tenant network
	Control & actions capabilities	**Description**	
12	Drop	Drop Packets	
13	SW Bandwidth	Set Minimum Bandwidth Warrantied and Maximum Bandwidth Available	
14	SW Latency	Set Traffic Priority (Directly associated to latency using PRIO queuing discipline)	
15	Set DSCP	Set Traditional IP DSCP field to signal/propagate QoS	
16	Inherit Overlay DSCP	Copy the DSCP value of the most inner overlay network to the outer IP DSCP field to signal/propagate QoS	
17	Forwarding	Set the interface where the packet will be forwarded	
18	Open Flow	Support for Open Flow protocol to allow SDN control capabilities	

capabilities are differentiated in two groups: classification and control capabilities, identified by a number that matches the number available in table II where a complete description of such capabilities is provided. These capabilities allow defining the classification of network slices based on devices, users, tenants or their respective services according to the flexibility and precision of the definition of network slices proposed in subsection II-B.

Capability 1 is required to support the *traditional best-effort IP network* scenario depicted in Fig. 1. Capabilities 1 and 2 are required to support the *service differentiation IP network* scenario. Capabilities 1 to 5 are required to support the *traditional LTE/4G network with QoS* scenario. Capabilities 1 and 2 and 8 to 10 are required to support the *multi-tenant infrastructure with QoS support*. Finally, capabilities 1 to 11 are required to support the *5G multi-tenant infrastructure with QoS support*. Netmap [14] and PF_RING [15] employ shared memory regions that can be mapped by both network devices and applications in user space and where network packets and their descriptors are stored. However, they do not provide any support for network functionalities such as QoS, traffic shaping or flow classification, and all these capabilities should be implemented from scratch in user space, as indicated in Table I. The Data Plane Development Kit (DPDK) [16] is a so-called *kernel bypass* framework that completely avoid the overhead introduced by the Linux kernel due to the complexity of handling a generic networking stack is avoided. Meanwhile, DPDK has maintenance and security drawbacks due precisely to this lack of Linux Kernel support and it has to re-implement in user space all the functionality provided by the kernel. Regarding classification capabilities, DPDK provides mechanisms to guarantee QoS and traffic shaping for network traffic that has been previously classified. However, current DPDK flow classification only supports three 4-tuple flow patterns for UDP, TCP and SCTP, respectively. In consequence, it lacks support for the different types of traffic available in a 5G multi-tenant infrastructure. The most widely tested software data path is the Linux Kernel Network Stack that provides two different frameworks to process packets: NetFilter [19] [20] and Linux Traffic Control (TC) [18]. Meanwhile, Netfilter lacks support to provide QoS and traffic shaping. On the contrary, the Linux Traffic Control (TC) does support QoS and traffic shaping but always in the context of the Tx Queues (transmission queues) of a given interface, and thus, it does not allow forwarding capabilities. Both Netfilter and TC can use either eBPF or u32 as a method for packet classification, and both provide similar capabilities to classify all the different types of traffic identified in Table I. However, to the best of our knowledge, there is no proposal of a software data path that explores these technologies for a 5G multi-tenant infrastructure to provide QoS and traffic shaping capabilities, probably due to the scalability limitations of the Linux networking stack. Salva et al. [21] is the only proposal found in the reviewed literature that is able to work

in a 5G multi-tenant network to process UHD video in 5G networks using Netfilter but focused on dropping packets in congested scenarios and not providing network slicing services with guaranteed QoS.

eXpress Data Path (XDP) [17] is a programmable packet processing platform that performs a high-speed packet processing up to 24 million pps. Nonetheless, XDP does not provide any QoS support or any classifier that allows the identification of 5G multi-tenant flows. Although such a classifier could be implemented using eBPF, there is no previous work providing such innovations.

In terms of control, capabilities 12 to 14 in Table I associated to the isolation of bandwidth, latency and the dropping of packets are the minimal ones required to provide network slicing in a concrete network segment. Thus, only DPDK, OpenVSwitch and TC are those that can provide such capabilities. When network slicing signaling is a requirement to consider scenarios where the slice needs to be extended across different network segments, capabilities 15 and 16 are required and this is where our contribution makes a differentiating point. It is noted that the DSCP field can be used for signaling QoS between different network devices. Our solution proposes a novel action related to the inheritance of the overlay DSCP signaling to pass the network slicing signaling information across the physical machines involved in the overlay networks and thus, ensures that such signaling is not discarded along the different network segments.

OVS [22] [23] is an open source, multi-platform software switch that has become the de-facto standard in virtualisation and SDN technologies. OVS offers support for the definition of QoS SLAs in terms of bandwidth and priority in traditional IP networks. However, it does not provide support for defining network slices, where SLAs are defined over the traffic available in overlay networks. In contrast, only our proposal, as indicated in Table I, provides all the classification and control capabilities required to support network slices for all the scenarios described, which is our main innovation.

Lastly, it is worth mentioning a complementary work focused on a different data path of the 5G network, the programmable hardware data path available inside of the network interface cards (NIC). Ricart-Sanchez et al. [24] propose a hardware-accelerated data path based on FPGA using the P4 language [25] to support network slicing in 5G multi-tenant traffic. Good performance is achieved due to the hardware approach but are limited in the number of rules supported, mainly due to hardware size constraints. Only 512 multi-tenant 5G network slices can be enforced simultaneously. Our software classification is inline with this hardware approach but ours is focused on providing network slicing capabilities for the virtual machines and their services with a significantly higher level of scalability with respect to the number of network slices supported.

IV. Proposed Network Slicing Support for 5G Multi-tenant Software Data Path

Fig. 3 shows a detailed view of the architecture proposed to support network slicing in the software data path of 5G

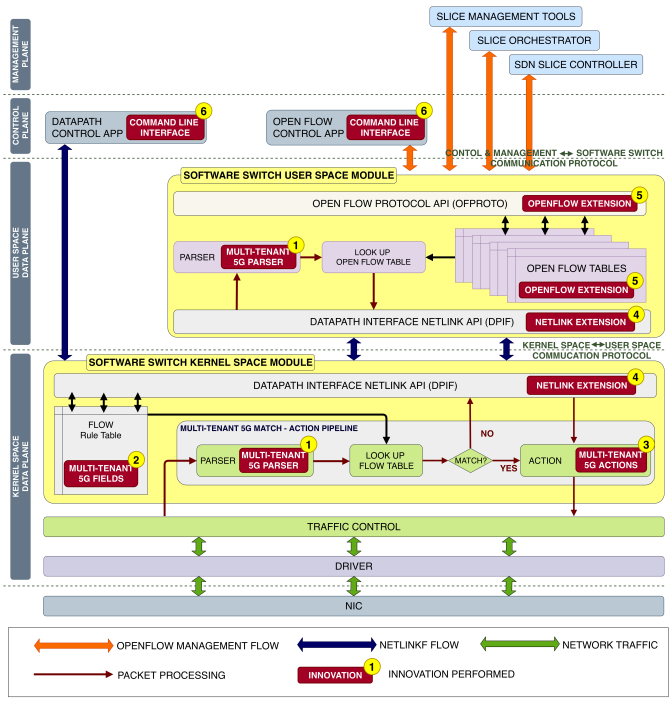

Fig. 3. Proposed software data path architecture with support for network slicing in 5G multi-tenant networks

multi-tenant networks. This architecture implements a SDN virtual switch that enables a dynamic and efficient network management in environments where virtualisation plays an important role such as 5G networks. The northbound interface that communicates with the SDN controller provides an OpenFlow API. OpenFlow [26] is considered the de-facto standard protocol SDN technologies. This architecture is based on a stack implemented in both the kernel and the user spaces to deal with scalability in terms of packet processing capabilities. Thus, the user space exposes OpenFlow capabilities (see upper part of fig. 3) where a SDN controller or simply network administrators can insert rules to program the behaviour of the data path. These rules are enforced in the kernel space only when there is active traffic that uses such rules, thereby maximising efficiency in hardware resources controlled by the kernel due to the fact that only a subset of rules is enforced at a given moment, depending on the current status of the communications. This approach is followed by OVS [23], employed as the base foundation to extend the network slicing capabilities presented in this work. Thus, in order to show clearly our contribution and innovations with respect to OVS, our proposed extensions are highlighted in Fig. 3.

When packets are received in the physical interface (see the bottom part of fig. 3), they are processed by the driver and sent to the traffic control module of the kernel. This traffic control module cannot perform any shaping of the traffic in the ingress interface since the traffic is physically there already and needs to be processed. The traffic is then inserted in our network slicing kernel module. The kernel module is a match-action pipeline where packets are classified to extract values that will be used later to be matched against the set of rules

installed to determine what action needs to be enforced to such packets. This packet classification should now be extended from the traditional classification for IP networks to a more complex classifier able to support all the data paths described in subsection II-B (see 1 in Fig. 3). This extension is explained in subsection IV-A. Moreover, the definition of the rules should also be extended to utilise the metadata extracted from the classifier (see 2). This is explained in section IV-B. If there is no rule matching the values extracted from the classifier, the packet is sent to the user space to determine if there are any installed rules there that need to be migrated to the kernel space, related to the packets being analysed. Thus, the protocol for communicating the kernel and the user space application needs to be extended to support the inter-exchange of such new information (see 4).

When the packets are received in the user space, they are parsed by another packet classifier to extract metadata. This extra classification process could be considered redundant but is introduced, with its overhead associated, to allow decoupling the user and the kernel spaces and allow them to work independently. Although the implementation details of this classifier are completely different to the classifier in the kernel, conceptually, they could be considered analogous and thus it also requires to be extended. Once the metadata is extracted, it is matched against the set of rules installed in the user space, usually referred to as OpenFlow tables. These tables need to be extended as well to allow the flexible definition of network slicing, including the most complex 5G multi-tenant network slices (see 5). If the metadata matches the rules stored, the packet is associated to the scope of a given network slice and thus, such rules are sent to the kernel space using the inter-exchange protocol, to be installed and executed therein (see 4). When the rule matches in the kernel, it means that the packet has been associated to a particular network slice. Thus, a set of actions can be applied over the packet to honour the QoS parameters specified in the definition of this network slice. This set of actions is described in section IV-C.

A. 5G Multi-tenant Classification Extensions

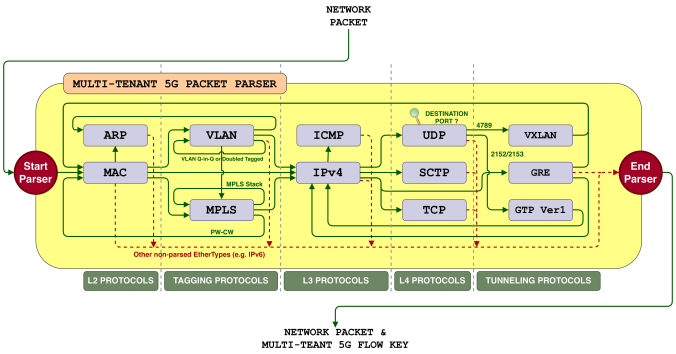

Fig. 4. Proposed flexible packet classification with support for 5G multi-tenant networks

Fig. 4 shows the architecture of the novel packet classifier designed to support all the data paths analysed. The parser can be followed from left to right where each of the steps is related to a layer of a protocol stack, including layer-2, tagging, layer-3, layer 4 and tunneling protocols. What makes this classifier different to traditional ones is the fact that it has arrows producing re-entrance in the classifier to continue the classification of the packets presented inside of the overlay networks. These capabilities significantly enhance the flexibility of the classifiers to support all the data paths. This is why it enables the flexible definition of network slices. It is worth noting how the key tunneling protocols for multi-tenant traffic isolation are present such as VLAN, VXLAN and GRE and how the key tunneling protocols for 5G user connectivity and mobility are present such as GTP. For traditional IP networks, the classifier only conducts one iteration on the parser. For 4G/5G networks and for multi-tenant networks, the classifier performs two different iterations. For 5G multi-tenant networks, the classifier runs three different iterations. The arrows going out of VXLAN, GRE and GTP in Fig. 4 illustrate what is an iteration of the classifier in this context. This new packet classifier has been prototyped in both the kernel and the user spaces (see 1 in Fig. 3).

B. 5G Multi-tenant Matching Extensions

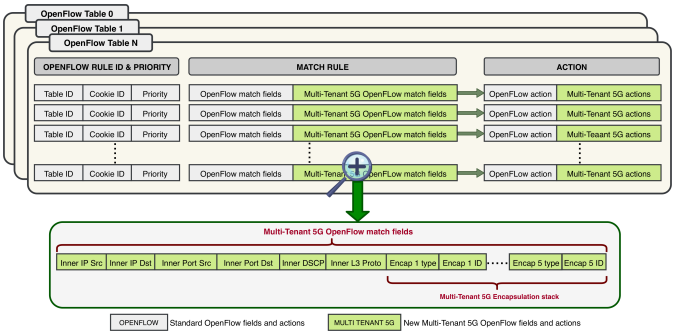

Fig. 5. Flow table structure with metadata extracted to support for 5G multi-tenant networks

Fig. 5 provides an overview of the structure used in the OpenFlow tables (see 5 in Fig. 3). It can be appreciated how there are different OpenFlow tables with an identifier used to assign priority in the processing of the set of rules available in the tables. Each table comprises a set of rules and each rule has an associated priority. Both rule priority and table priority determine the global processing order of such a rule. Each rule is a traditional OpenFlow match-action rule. Different metadata fields are extracted in each of the steps of the classifier to conform the set of metadata values that will be checked in the match part of the rule. These key metadata values have been significantly extended to support the flexible definition of network slices in the overlay networks. Hence, the traditional 6-tuple (*src ip, dst ip, l4 protocol, src port, dst port, dscp value*) has been extended to deal with the support for overlay networks associated to 5G and multi-tenant networks. To do so, the 6-tuple is utilised to match the traditional IP traffic and the metadata has been extended with another analogous 6-tuple created to match the fields of the overlay networks (*inner src ip, inner dst ip, inner L4 protocol, inner src port, inner dst port, inner dscp value*).

Meanwhile, this extension imposes challenges to be addressed. Let us focus on a simple scenario where the data path does not contain any nested overlay networks as those imposed by 5G multi-tenant networks (see E in Fig. 1) and the data path only has multiple non-nested overlay networks (see C and E in Fig. 1). To define a rule where the inner 6-tuple of the overlay network matches against a particular overlay network, such a network needs to be identified. This is why the metadata has been extended with a field (*encapsulation id*) to identify such a network. Then, a problem arises when the data path is used to support multiple encapsulation protocols simultaneously. For example, a scenario where both support for a multi-tenant infrastructure and for a 4G/5G network need to be provided at the same time. In the hypothetical scenario where the encapsulation id used to identify the overlay network of a particular tenant, it could collide with the encapsulation id used to identify a user in a 4G/5G network. The rule will match, causing an incorrect application of the network slice. In such scenarios, there is a need to match the metadata associated to the encapsulation protocol utilised to implement the overlay network. To this end, another field entitled (*encapsulation type*) has been added. This field checks the protocol utilised to implement the overlay network. Thus, it differentiates "vxlan" and "gtp" for the above example described. Currently, this field takes 5 values: "mpls, vlan, vxlan, gre, gtp", for the proposed parser.

At this point, the definition of network slices is supported for multiple concurrent non-nested overlay networks. To extend such support to deal with nested overlay network, the maximum possible levels of nested overlays should be defined. In our prototype, it has been fixed to five levels, based on the rationale in the most complex scenario that would make sense in 5G infrastructures. These five levels comprise one overlay for the management of physical networks, one for the management of the virtual/tenant networks, one for the management of the vertical/5G user networks, one for the management of DPI packet dissection and one additional one for future extensions. It is worth mentioning that there is a trade-off between overhead in the matching time and flexibility in the support provided for different data paths. Thus, the previous two new fields are repeated five times, leading to (*encapsulation type 1, encapsulation id 1, ..., encapsulation type 5, encapsulation id 5*).

However, a new challenge needs to be addressed. When there is a nested overlay network, the parser will populate the 6-tuple for traditional IP networks, the 6-tuple for overlay networks with the information of the first overlay network, and then when the parser processes the second overlay network, it will override the values of the 6-tuple for overlay networks with the new information, and so on. As a consequence of this overriding, the 6-tuple for overlay networks will have information only about the innermost overlay network and the information for the intermediate ones will be overridden. The alternative to overcome this limitation is to replicate five times the 6-tuple, one per each different nested overlay networks supported. That extension has been carefully considered but it imposes a significant impact on performance, overhead and scalability of the proposed approach. Thus, a trade-off has been decided based on the selection of the intermediate traffic using the encapsulation type and encapsulation id fields whereas for the final overlay network it can also use the information of the 6-tuple associated. Thus, as a result, the metadata extracted is composed by 2 x 6-tuples plus 5 x 2-tuples, totaling 22 fields to check in the matching stage. The matching stage supports the use of wild cards to enable or disable totally or partially the matching of such a field. The extension over the traditional fields is indicated in the lower part of Fig. 5.

The structure of metadata of the proposed OpenFlow tables are conceptually similar to the structure of the flow tables available in the kernel space (see 5 in Fig. 3). However, there are some difference. Firstly, rules in the kernel space do not support wildcards. Thus, a conversion process of one to many is conducted in the rules installed in the kernel space in order to convert the wild card to the possible combinations to be matches in the kernel. Secondly, rules in the kernel space are only stored for a given time and then they expire from the table if there is no traffic matched.

C. Network Slicing Action Extension

When a network slice matches a rule, a set of actions needs to be applied to enforce performance isolation of traffic among network slices. These actions are listed as follows: i) a warranted minimum bandwidth available for this slice, ii) a maximum allowed bandwidth available for the traffic of this slice, and iii) a priority to the traffic of this slice. The priority has a direct impact on both latency and packet loss. Thus, the higher the priority, the lower the delay and packet loss. These three primitives allow network performance isolation among slices and apply to all the data paths described (Capabilities 12 to 14 shown in Table I). These primitives allow enforcing performance isolation in network slices at a particular point of the network. These are the extensions depicted as 3 in Fig. 3. To this respect, there is no need to propagate the maximum bandwidth allowed along different network segments since it can be enforced in the perimeter of the network and will be always respected along the data path since the specific traffic does not grow (noticeably) in size during transmission.

The priority, instead, needs to be signalled along the data path to allow different data path components to be aware of the priority. In traditional IP networks, DSCP is used for this purpose (Capability 15 in Table I). However, for the novel 5G multi-tenant networks, the packet sent by a 5G user changes its shape along the data path every time the packet enters or leaves an overlay network due to the encapsulation/decapsulation process associated. Thus, a mechanism based on the copy of the DSCP value from inner to outer headers (COPY_INNER_TOS) and vice versa (COPY_OUTER_TOS) is implemented to keep these values even when the encapsulation is removed and thus allowing the signalling across the whole data path (see 3 in Fig. 3).

The queuing discipline implemented to enforce the warranty of both bandwidth and delay is based on a Hierarchical Token Bucket (HTB) with a bucket of a limited maximum bandwidth only and a minimum warranted bandwidth matching the maximum speed of the interface. Different children

token buckets are attached to the parent to allow borrowing capacity across all the children. Then, one child is attached per network slice. The children are defined by three values: limited maximum bandwidth, minimum warranted bandwidth, and priority. When they are not explicitly defined, the system will assume the default value 0. The attachment of a new child HTB triggers a new check to see if the minimum warranted bandwidth can be fulfilled and inform the error to the management plane.

When packets arrive at an interface, they are assigned to their associated slice by mean of a rule defined in the software data path, appended to the associated child HTB to this slice. If there is no rule matched, by default the traffic is assigned to slice 0. The packets are then policed against the bandwidth limitations and after that are de-queued in strict prioritisation with respect to the priority number associated to this child HTB. This mechanism allows performance isolation of network slices in terms of bandwidth and delay.

D. 5G Multi-tenant Protocol Extensions

The NetLink protocol in Linux kernel for kernel-user space inter-exchange communications has been extended to allow inter-exchanging the new metadata in the rules. When rules are moved from the user space to the kernel space, a Cartesian product needs to be applied over wildcard values in the rules, producing different rules to be implemented in the kernel space. This mechanism has been extended with the new fields proposed (see 4 in Fig. 3). Consequently, the OpenFlow protocol for matching the new fields and actions has also been extended (see 5 in Fig. 3). Finally, the command line tools to allow administrators to insert such slice definitions in both OpenFlow tables in the user space and the kernel tables have been prototyped (see 6 in Fig. 3).

E. 5G encapsulation/decapsulation and fine-grained network slicing capabilities

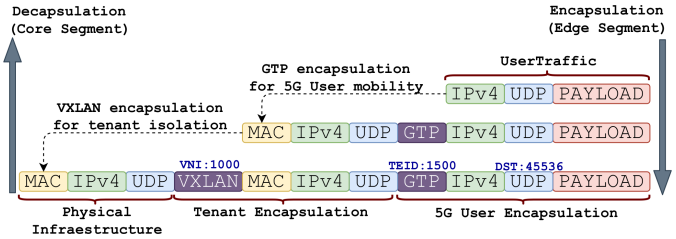

Fig. 6. Double encapsulation in 5G networks for tenant isolation and user mobility

Two critical features of 5G Networks are the control of user mobility between different gNBs and the traffic isolation between tenant networks that are sharing the same physical infrastructure. On the one hand, user mobility is achieved by encapsulating the user's original traffic with the GTP protocol. On the other hand, tenant isolation is provided by using tunneling protocols such as VXLAN, GRE, GENEVE or STT. Figure 6 illustrates an example of encapsulation where the VXLAN protocol is used to create the tunnels for each tenant. It is worth to emphasize that from a network slicing perspective, despite of the fact that the three packets shown in figure 6 have a different packet structure, they are the same user's traffic that is changing its shaping across the different network segments in the E2E data path. The GTP encapsulation and decapsulation is performed by the gNB in the edge segment and by the UPF in the core segment. The virtual switches that interconnect the virtual machines on both edge and core segments are responsible of the overlay encapsulation and decapsulation for tenant isolation (see figure 2 in subsection III-A).

Therefore, the first stage to provide fine-grained network slicing capabilities is to be able to identify these types of traffic within a 5G network and access the inner headers with the information about the user and its associated metadata (GTP TEID, VNI, etc.). This is the reason behind the design of the parser (see subsection IV-A) and the new fields added to the match-action rules (see subsection IV-B). Our approach does not propose any particular method to carry out encapsulation and decapsulation. The method used to encapsulate/decapsulate traffic does not affect our implementation and does not require any special customization. Our innovation comes from the possibility to provide support for network slicing in such complex 5G traffic.

The following command line provides an example where the video flow coming from the 5G ambulance to the hospital need to be prioritized to guarantee patient care while in the road. In summary, all the video traffic to the destination port 45536 from the paramedic (user with GTP TEID 1500) belonging to the virtual mobile network operator with VXLAN ID 1000 is mapped to the slice with id 4 and sent out to the port *vp2* to the hospital with a very high priority (5). The reader can appreciate how this rule carry out the matching of the double encapsulated traffic according to the values indicated in the bottom part of Figure 6.

```
root@host:~$slicenetvswitch add-flow br0
   table=0,priority=5,in_port(vp1),eth()
   tunnel_type_1=VXLAN, tunnel_id_1=1000,
   tunnel_type_2=GTP, tunnel_id_2=1500,
   inner_dst_port = 45536,
   actions=set_slice_id(4),vp2
```

V. Implementation Details and Empirical Results

In this section, we first provide details on the implementation of the proposed Software datapath. Then, we demonstrates the suitability and scalability of the new 5G multi-tenant network slicing capabilities prototyped in *SliceNetVSwitch*, ensuring isolation among network slices and providing mechanisms to deliver guaranteed QoS in terms of delay and bandwidth for each network slice. It also provides analysis of the overhead in performance introduced by the novel functionalities, and scalability in the number of network slices supported and how many flows can be attached to such slices.

A. Implementation Details of Network Slicing Support

The proposed software data path has been prototyped with significant extensions to OVS version 2.9.2. To achieve the

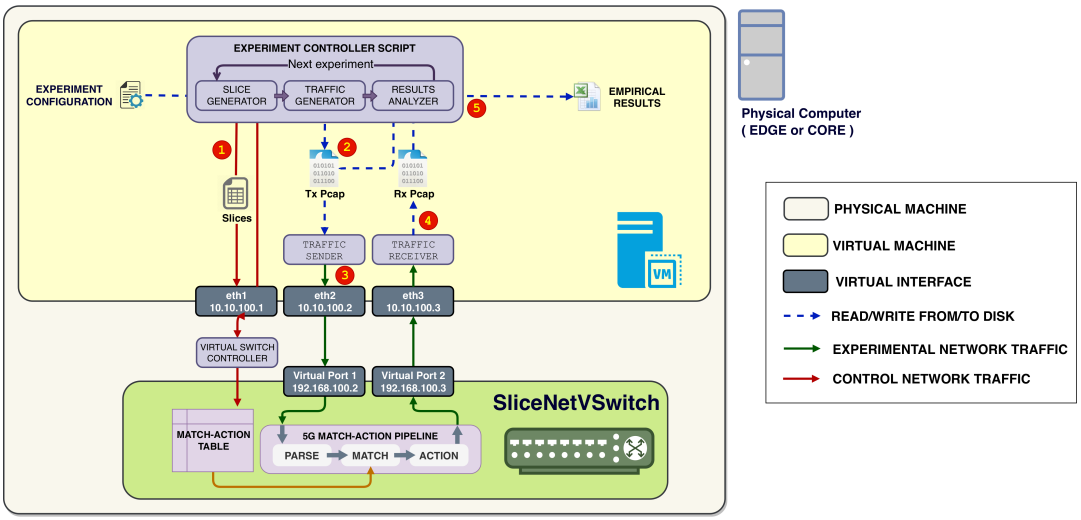

Fig. 7. Testbed deployed for 5G software data path and empirical results analysis

new network slicing functionalities over 5G multi-tenant networks, we have extended the *openvswitch.ko* kernel module, the *vswitchd* user space daemon, and the *ovs-ofctl* and *ovs-dpctl* command line utilities in order to match the proposed architecture. The extensions are enabled and disabled in the configuration step of the compilation process of OVS by indicating new flags, as a plug-in. OpenFlow v1.4 of the Northbound Interface of OVS has also been extended with the newly proposed fields to enable the new flexibility in the proposed various definitions of network slices. The extended version of OVS is named *SliceNetVSwitch*. The data structures that contain the flow rules have been extended with the new fields that allow the definition of 5G network slices (see IV-A and IV-B). These data structures are different in user space and kernel space and hence it has been needed to modify them in both modules. The source code of the *dpif* library has been extended. This library is in charge of enabling communication between the user space daemon and the kernel module using the *netlink* protocol. The format of the *netlink* messages has been extended as well as the code needed to process them. Analogously, the *ofproto* module within the user space daemon has also been enhanced. This module handles all related to the OpenFlow protocol. These new OpenFlow extensions to the fields and actions extend the expressiveness of the programmability primitives exposed to the user. This new expose has a twofold implication. First, it allows users to program the definition of the slices in the 5G network. Second, it allows the new API to be integrated in Software-Defined Network architectures, being the main pillar of the 5G networks. The new OpenFlow Extension provide a fine-grained mechanism for handling network slicing with 5G capabilities to the control and management layer.

B. Experimental Setup

All the experiments were executed on a computer with the following hardware specifications: Dell T5810 with an Intel Xeon E5-2630 v4 CPU, 10 cores with hyper-threading, 32768 MBytes of RAM and 512 GByte SSD hard disk. This computer is either an edge or a core node in the proposed 5G architecture with the proposed *SliceNetVSwitch* installed. Fig. 7 shows the testbed deployed to conduct the empirical validation of the new capabilities of the prototype. The testbed is driven by the experiment script controller that sets up each experiment with different parameters. The different parameters analysed and their ranges are detailed in Table III.

Firstly, for each scenario , the experiment script controller configures the software switch to inject the configurations about the slices on-boarded into the system and the associated flows to each of such slices, through a dedicated management interface (see 1 in Fig. 7). Secondly, the traffic generator agent (TGA) generates several PCAP files, which are then sent in parallel by the traffic sender agent (2 and 3 in Fig. 7). This traffic is generated by the TGA following the experiment parameters (number of tenants, number of users per tenant, number of services, number of slices, Tx bandwidth per slice, etc.). The *SliceNetVSwitch* receives and processes the traffic in the novel match-action pipeline, i.e., parses of the packet, looks up in the match-action table and performs the slice mapping. The outgoing traffic is sent back for the purpose of being captured by the traffic receiver agent and saved in a Rx PCAP file (4 in Fig. 7). Finally, the results analyzer agent compares both generated PCAP files, received and sent, to obtain the results of the experiment (delay, jitter, bandwidth). Every sent packet is identified by a unique 8-byte ID that is included in the payload. Every packet is captured twice during the transmission. First, when it is sent by the Traffic Sender. And second, when it is received by the Traffic Receiver. In both cases, the packet is saved in a PCAP file with a timestamp. The delay is calculated as the difference between both timestamps and the ID is used to unequivocally identify each packet. For performance evaluation purposes, traffic is sent and received in the same node to provide full time synchronization between sender and receiver and thus allowing to measure an accurate latency at nanosecond scale.

TABLE III
RANGE OF VALUES FOR EACH PARAMETER IN THE EXPERIMENTAL TESTBED

Parameter	Values	Minimum	Maximum
Packet size	64, 128, 256, 512, 768, 1024, 1280, 1500	64	1500
Bandwidth (Mbps)	1000, 5000, 10000, 15000, 20000	1000	20000
Number of Rules	1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536	1	65536
Number of Slices	1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4196, 8192	1	8192
Type of Traffic	IP, 4G/LTE, Multi-Tenant, 5G Multi-Tenant	IP	5G Multi-Tenant

TABLE IV
DIFFERENT TYPES OF TRAFFIC USED IN THE TESTBED

Type of Traffic	Number of Headers	Number of Encapsulations	Total Headers Length (in Bytes)	Packet structure
IP	4	0	42	MAC/IP/UDP/PAYLOAD
LTE/4G	6	1	82	MAC/IP/UDP/GTP/IP/UDP/PAYLOAD
Multi-Tenant	7	1	92	MAC/IP/UDP/VXLAN/MAC/IP/UDP/PAYLOAD
5G Multi-Tenant	10	2	132	MAC/IP/UDP/VXLAN/MAC/IP/UDP/GTP/IP/UDP/PAYLOAD

To empirically validate the proposed architecture, four different types of traffic were generated according to the different scenarios described in subsection II-B: Traditional IP, LTE/4G, Multi-Tenant and 5G Multi-Tenant traffic. Each kind of network traffic has different number of headers. This introduces an overhead in the parsing (a deep packet dissection is performed) and matching (the key has been expanded with new fields) phases: the more complex the packet, the longer it takes to process it. Table IV shows all the parameters that were ranged in the different experiments.

C. Overhead in Network Performance

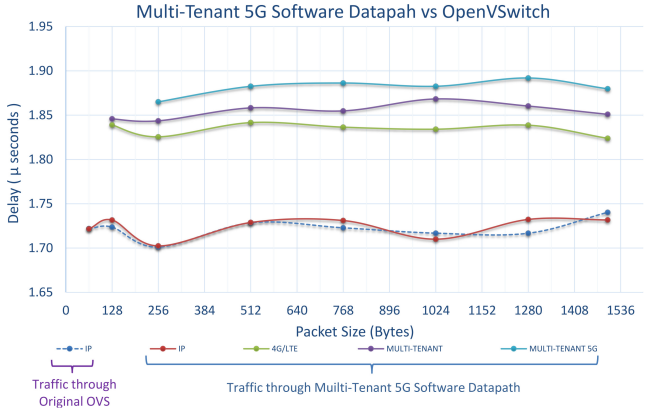

Fig. 8. Overhead of the proposed 5G multi-tenant software data path for different types of traffic and MTUs when compared to original OpenVSwitch

Firstly, we analysed the cost of adding the new functionalities. At 20 Gbps, Fig. 8 shows the average delay introduced by the proposed *SliceNetVSwitch* when compared with the original OVS implementation by ranging the packet size. The data shown corresponds to the average of the packet processing time of the traffic transmitted along 10 seconds. Preliminary tests revealed that once traffic is steady (less than 0.3 seconds in the worst case), it remains constant in terms of delay, jitter and packet loss and thus 10 seconds is time enough to produce accurate results. For traditional IP traffic, the behaviour was fairly similar across both implementations, indicating that the extensions introduce almost no delay, which clearly provides significant efficiency in the implementation of the new capabilities. For the most complex kinds of network traffic, the original OVS cannot be compared since it does not support such capabilities. However, it can be noticed how much extra delay is associated to the extra complexity associated to each kind of traffic. Thus, the 5G multi-tenant traffic, which is the most complex traffic to process, imposed an additional overhead of about 0.2 microseconds (us) when compared with traditional IP traffic. The absolute values were around 2 us, which is very acceptable. In all the experiments, there is not any packet loss in both OVS and our solution and we are gathering the same throughput for the sender and receiver, thus there is not any impact in the throughput.

D. Calculating Safe Boundaries

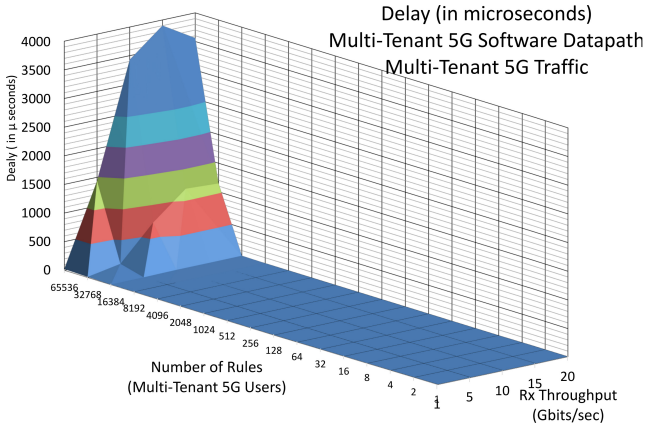

Fig. 9. Analysis of delay over 5G multi-tenant traffic for the prototyped software data path (scalability with respect to the number of rules and Rx bandwidth).

When implementing network slicing, the delay and bandwidth should be warranted within the same slice and isolated from other slices. Thus, it is important to understand how many network slices can be supported in the system before it starts to violate such warranty due to lack of resources. To this end, Fig. 9 and Fig. 10 show an analysis where both delay

and packet loss were respectively analysed when different transmission speeds and the number of rules were ranged linearly and exponentially, respectively. This scenario sent 1500-byte packets and there was only one slice on-boarded in the system, defined in terms of limited maximum bandwidth (20 Gpbs), minimum warranted bandwidth (20 Gbps) and priority (1). 20 Gbps were sent for 10 seconds of the most complex traffic (5G multi-tenant traffic). Priority was irrelevant since there was only one slice. Then, there were as many flows as the number of rules being inserted in *SliceNetVSwitch*. All these flows were attached, one by one, to the same slice, and thus should be treated equally. Both Fig. 9 and Fig. 10 show the difference in their treatment. As shown, up to 16384 network flows were assigned, one by one, to the same network slice; all of them were respected in terms of delay and packet loss at all the transmission speeds analysed, up to 20 Gbps. It clearly indicates the boundaries that can be used in terms of the management of network slices and on the attachment of flows to network slices before the system starts to violate the warranted performance due to the scalability of the architecture. It is noted that 16364 network flows at 20 Gbps with 16364 rules installed in our architecture to warrant the QoS produced 0% of packet loss and a delay of 9.34 us. These results demonstrate the high scalability of the proposed architecture. Graphs for bandwidth received are not presented since it can be inferred from Tx bandwidth minus packet loss.

Similar executions were carried out with the other types of traffic in order to validate the flexibility of the proposed architecture. In all the cases, the boundaries of scalability showed similar results, in terms of around 16364 rules. Thus, these similar graphs were not presented for brevity. The following is an example of the command line command used to introduce the network slice utilised in this experiment where a slice with id *slice-id-1* was created in port *vport1* with a minimum warranted bandwidth of 10 Mbps, a maximum bandwidth of 12 Mbps and priority 0 (the highest priority). The bandwidth of the interface was limited to 1 Gbps:

```
root@host:~$slicenetvswitch set port port1
    set port port1 qos=@slice1
    --id=slice1 create qos type=linux-htb
    other-config:max-rate=1000000000
    queues:1=@slice-1
    --id=@slice-1 create slice
    other-config:min-rate=10000000
    other-config:max-rate=12000000
    other-config:priotity=0
```

E. Empirical Validation of Network Slicing

In order to empirically validate the effectiveness of the network slices, we executed scenarios when ranging exponentially the number of slices between one and 16384. It is noted that 16384 had been empirically achieved in the results previously analysed. In this experiment, there was one network flow associated to each of the available slices. Each slice had a different priority ranging from 0 to 16384 by increment of one. They also had the same maximum limited bandwidth. It was the total bandwidth of the experiment divided by the total number of slices so that they were equally split among them.

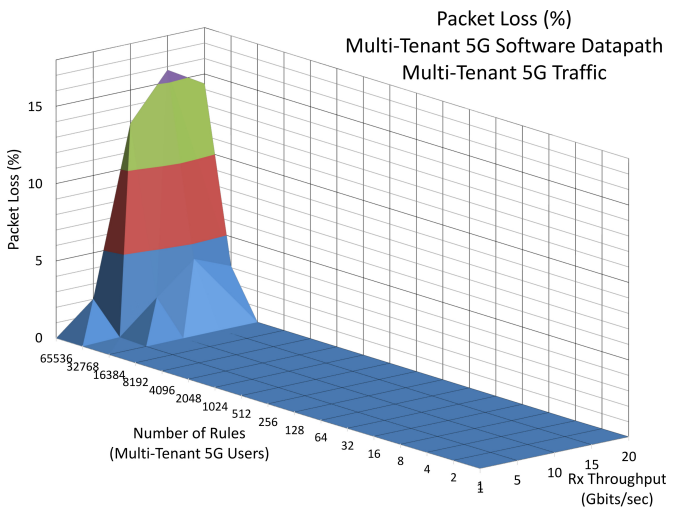

Fig. 10. Analysis of packet loss ratio over 5G multi-tenant traffic for the prototyped software data path (scalability with respect to the number of rules and Rx bandwidth).

In terms of the minimum bandwidth warranted, it was setup as 100% of the maximum limited bandwidth.

This means 30 scenarios (15 scenarios ranging from 1 to 16384 slices at 20 Gbps and 15 at 10 Gbps) where both delay and packet loss should be plotted, totaling 60 graphs. For each of these graphs, it should be drawn as many series as the number of slices associated to the graph. An intensive analysis has been carried out to deal with this amount of information. For the most complex scenario where there are 16384 network slices and 16384 flows, totaling 20 Gbps, it has been empirically demonstrated that delay of all the slices were not respected under this level of stress. The reason now is completely different. The HTB queuing discipline in the Linux kernel does not support to efficiently deal with 16384 different queues, required to control all the priorities associated to each of slices, at 20 Gbps. The most complex scenario that validated the expected behaviour in terms of the performance isolation across network slices was 8192 network slices and 8192 flows, totaling 10 Gbps. Smaller scenarios showed similar trends.

In these scenarios, there was 0% of packet loss in the whole experiment. It means that all the packets sent were received. Thus, it is irrelevant to include a graph about bandwidth and packet loss. Instead, we plotted a heat map for the analysis of delay. The network slice identification number in the Y-axis allows understanding that all the heat information available horizontally corresponds to each of the slices deployed in the data path. The X-axis allows observing how each of the network slices behaved along the duration of the experiment. This experiment sent 0.83 millions of packets per seconds of a size of 1500 bytes (i.e., at 10 Gbps). The network slice identification corresponds to the priority associated to that slice. Thus, in the analysis of delay, if the proposed network slicing control was working as expected, the heat map should gradually be increasing color from bottom to top due to the fact that slices with higher priority are faster than those with lower priority. At the same time, the heat map kept a constant color when reading the heat map horizontally across

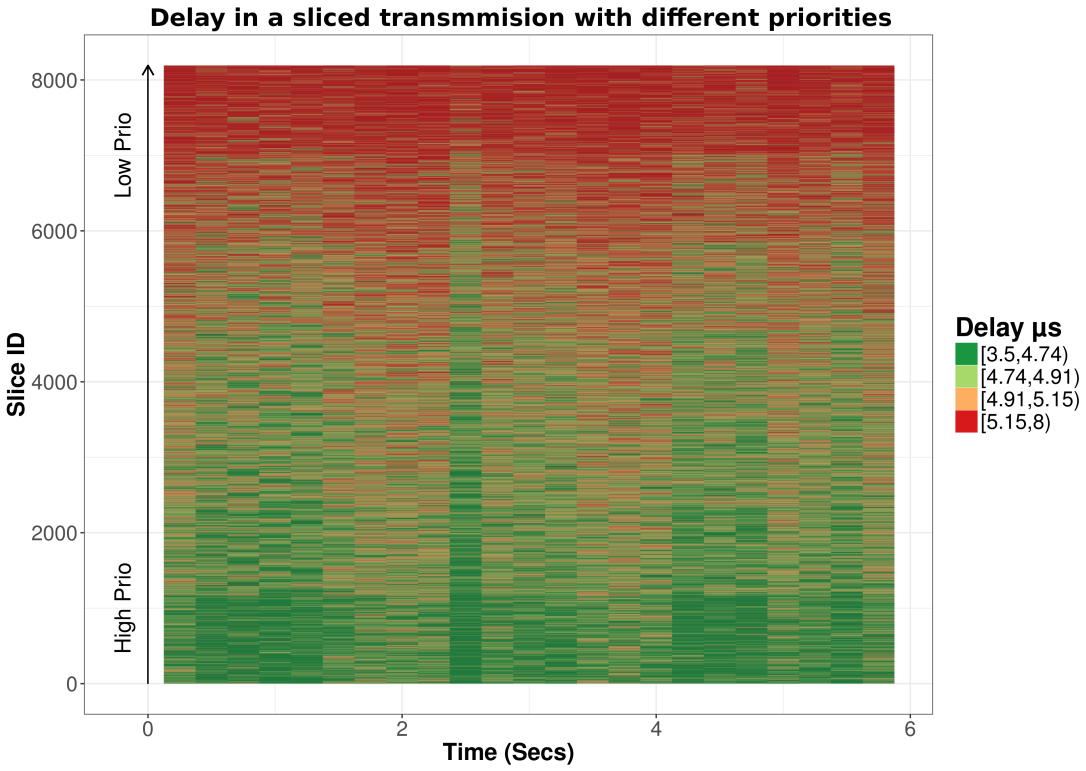

Fig. 11. Average delay by slice in the scenario with 8192 slices where all slices were transmitting with the same bandwidth with a total throughput of 10Gbps during 6.5 seconds

the same slice. This behaviour perfectly matches the empirical results obtained in Fig. 11 where delay is shown for the 8192 slices. When reading the graph horizontally, one can notice some variations in delay along the same slices, mainly due to jitter. The important aspect is to see that such variations were reduced when the priority of the slice was increased, achieving an almost constant behaviour for the most prioritised slices. The heat map was divided in four different intervals, corresponding to four different quarterlies for all the delays analysed. The percentage of packet going to the user space with respect to the total number of packets processed in the kernel space for the worst-case scenario of our experiments, at 10 Gbps and at biggest packet size, i.e. 1500 bytes, generates "only" 0.8 Mpps, and from them, 8192 packets (1 per flow) are going to user space. Thus, 0.01% of packet are going to user space in the worst-case scenario. With respect to the uRRLC 5G ambulance use case, the traffic will be mapped to a high priority slice (one of the green slices at the bottom of the heat map). Therefore, it will allow to offer a reliable and guaranteed service with low latency to make sure the patient has the better possible care. Furthermore, in a congested network that traffic will have priority over other slices, with lower priority, and thus, the bandwidth will also be guaranteed.

Fig. 11 has more than 204800 data points after averaging the 5600000 data points (one per packet) on time intervals (0.25 seconds). There is only one tool, the statistical framework *R*, that is able to render this graph from more than 10 different tools analysed. When absolute values are compared between Fig. 9 and Fig. 11 for the case with one and 8192 slices at 10 Gbps, respectively, it can appreciated how the overhead in delay to deal with only one slice was around 4.48 us whereas the delay associated to 8192 slices was around 4.98 us. These results have validated the high scalability in terms of the number of slices and the fact that the number of slices up to 8192 only affected 500 nanoseconds, which is insignificant for a software-based solution. Consequently, the system has demonstrated that it is able to handle 8192 slices with guaranteed delay, bandwidth and packet loss.

VI. Conclusion and Future Work

An unambiguous and flexible definition of network slicing suitable for the novel 5G multi-tenant infrastructures has been provided in this paper. This definition has been used as a fundamental part for the design of new network slicing control capabilities that allow warranting QoS in terms of bandwidth and delay for network slices comprising network flows, that are flexibly defined, with expressiveness enough to allow the definition of complex 5G multi-tenant traffic and network slices, far beyond the DSCP services of the traditional TCP/IP network stack. The novel network slicing capabilities have been successfully prototyped and a complete view of the architecture of this prototype has been provided. Empirical results indicate that the implemented prototype is able to deal with up to 8192 network slices at up to 10 Gbps. The performance isolation under this level of stress is respected across all the traffic associated to each of the slices and also across different network slices. It is worth highlighting the superior results associated to this number

of slices. The proposed prototype has demonstrated that the new capabilities implemented do not impose any delay with respect to a reference software (OVS) that does not support such capabilities. Even in the most stressful scenario (worst case), the maximum delay is around 5.8 us and the maximum packet loss is around 0%, which are within the scope of the boundaries of the performance of 5G networks, which has clearly validated the suitability of the proposed architecture.

As a future work, we will investigate different 5G use cases such as NB-IoT which requires the network to control a million devices per square meters in the software data path. Moreover, future work will seek to accelerate some of the processes on the software data path in hardware by offloading selected processing tasks. Finally, much higher transmission speeds (up to 100 Gbps) will be explored by employing kernel bypass technologies in order to study if the scalability of the number of slices can be improved by some orders of magnitude, allowing ultra-fine-grained control of the network.

ACKNOWLEDGMENT

This work was funded by the European Commission under Grant Agreement H2020-ICT-2016-2/761913 SliceNet (End-to-End Cognitive Network Slicing and Slice Management Framework in Virtualised Multi-Domain, Multi-Tenant 5G Networks) and by the UWS 5G Video Lab project.

REFERENCES

[1] P. e. a. Popovski, "5G wireless network slicing for eMBB, URLLC, and mMTC: A communication-theoretic view," *IEEE Access*, vol. 6, no. 648382, pp. 55 765–55 779, 2018.

[2] C. Simon, M. Maliosz, J. Bíró, B. Gerő, and A. Kern, "5G exchange for inter-domain resource sharing," in *2016 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, 2016, pp. 1–6.

[3] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "NFV and SDN-Key technology enablers for 5G networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2468–2478, 2017.

[4] I. e. a. Afolabi, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.

[5] C. Bouras, P. Ntarzanos, and A. Papazois, "Cost modeling for SDN/NFV based mobile 5g networks," in *2016 8th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*. IEEE, 2016, pp. 56–61.

[6] S. e. a. Hu, "Providing Bandwidth Guarantees, Work Conservation and Low Latency Simultaneously in the Cloud," *IEEE Transactions on Cloud Computing*, vol. PP, no. c, p. 1, 2018.

[7] P. e. a. Rost, "Network Slicing to Enable Scalability and Flexibility in 5G Mobile Networks," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 72–79, 2017.

[8] S. Zhang, "An Overview of Network Slicing for 5G," *IEEE Wireless Communications*, vol. PP, pp. 1–7, 2019.

[9] NGMN Alliance, "NGMN 5G P1 Requirements & Architecture Work Stream: End-to-End Architecture Description of Network Slicing Concept," pp. 1–11, 2016. [Online]. Available: https://www.ngmn.org/wp-content/uploads/Publications/2016/161010_NGMN_Network_Slicing_framework_v1.0.8.pdf

[10] GSM Alliance, "An Introduction to Network Slicing," 2017. [Online]. Available: https://www.gsma.com/futurenetworks/wp-content/uploads/2017/11/GSMA-An-Introduction-to-Network-Slicing.pdf

[11] A. Galis, "Network Slicing Terms and Systems," 2017. [Online]. Available: https://datatracker.ietf.org/meeting/99/materials/slides-99-netslicing-alex-galis-netslicing-terms-and-systems

[12] A. e. a. De La Oliva, "5G-TRANSFORMER: Slicing and orchestrating transport networks for industry verticals," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 78–84, 2018.

[13] J. Kim, D. Kim, and S. Choi, "3GPP SA2 architecture and functions for 5G mobile communication system," *ICT Express*, vol. 3, no. 1, pp. 1–8, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.icte.2017.03.007

[14] L. Rizzo, "Netmap: a novel framework for fast packet I/O," in *21st USENIX Security Symposium (USENIX Security 12)*, no. 257422, 2012, pp. 101–112. [Online]. Available: https://www.usenix.org/conference/usenixfederatedconferencesweek/netmap-novel-framework-fast-packet-io

[15] ntop, "PF RING Documentation web site." [Online]. Available: https://www.ntop.org/guides/pf_ring/

[16] The Linux Foundation Projects, "Data Plane Development Kit Project (DPDK)." [Online]. Available: https://doc.dpdk.org/

[17] T. e. a. Høiland-Jørgensen, "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*, ser. CoNEXT '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 54–66. [Online]. Available: https://doi.org/10.1145/3281411.3281443

[18] The Linux Documentation Projects, "Linux Traffic Control (TC)." [Online]. Available: http://tldp.org/HOWTO/Traffic-Control-HOWTO/index.html

[19] Harald Welte, Pablo Neira Ayuso, "The Netfilter project." [Online]. Available: https://www.netfilter.org/

[20] R. Rosen, *Netfilter*. Berkeley, CA: Apress, 2014, pp. 247–278. [Online]. Available: https://doi.org/10.1007/978-1-4302-6197-1_9

[21] P. Salva-Garcia, J. M. Alcaraz-Calero, R. M. Alaez, E. Chirivella-Perez, J. Nightingale, and Q. Wang, "5G-UHD: Design, prototyping and empirical evaluation of adaptive Ultra-High-Definition video streaming based on scalable H.265 in virtualised 5G networks," *Computer Communications*, vol. 118, pp. 171 – 184, 2018.

[22] B. P. et alt., "The Design and Implementation of Open vSwitch," in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. Oakland, CA: USENIX Association, May 2015, pp. 117–130. [Online]. Available: https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/pfaff

[23] The Linux Foundation Project, "Open vSwitch website," 2019. [Online]. Available: https://www.openvswitch.org/

[24] R. Ricart-Sanchez, P. Malagon, A. Matencio-Escolar, J. M. Alcaraz Calero, and Q. Wang, "Toward hardware-accelerated QoS-aware 5G network slicing based on data plane programmability," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 1, p. e3726, 2020, e3726 ett.3726. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3726

[25] P. e. a. Bosshart, "P4: Programming protocol-independent packet processors," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014.

[26] OpenFlow Networking Foundation, "Openflow switch specification version 1.5.1 (protocol version 0x06)," 2015. [Online]. Available: https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf

Antonio Matencio Escolar Antonio Matencio Escolar is a PhD candidate at the University of the West of Scotland in UK. He is actively involved in the H2020 5G-PPP Slicenet project. His main research interests include network slicing, software datapath, SDN, 5G mobile networks and network control and management, among others.

Qi Wang Qi Wang is a Full Professor at the University of the West of Scotland, and he is the technical co-coordinator of EU H2020 5G-PPP Phase I SELFNET and Phase II SliceNet. He is a Board Member of the Technology Board of EU 5G-PPP. His research primarily focuses on 5G mobile networks and video networking. He has a PhD in mobile networking from the University of Plymouth, UK.

Jose Alcaraz Calero Jose M. Alcaraz-Calero is a Full Professor in networks and security at the University of the West of Scotland, and he is the technical co-coordinator of the EU H2020 5G-PPP Phase I SELFNET and Phase II SliceNet. His professional interests include network cognition, management, security and control, service deployment, automation and orchestration, and 5G mobile networks. Alcaraz Calero has a PhD in computer Science, University of Murcia