IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 8, 2021, accepted December 6, 2021, date of publication December 15, 2021,
date of current version December 27, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3135827

Performance Isolation for Network Slices in Industry 4.0: The 5Growth Approach

CHIA-YU CHANG[1], (Member, IEEE), TERESA GINER RUIZ[2], FRANCESCO PAOLUCCI[3], MANUEL A. JIMÉNEZ[4,5], JAVIER SACIDO[2], CHRYSA PAPAGIANNI[1,6], (Member, IEEE), FABIO UBALDI[7], DAVIDE SCANO[8], MOLKA GHARBAOUI[3], ALESSIO GIORGETTI[8,9], LUCA VALCARENGHI[8], (Senior Member, IEEE), KONSTANTIN TOMAKH[10], ANDREA BODDI[7], AGUSTÍN CAPARRÓS[2], MATTEO PERGOLESI[2], AND BARBARA MARTINI[3]

[1]Nokia Bell Labs, 2018 Antwerp, Belgium
[2]Telcaria Ideas S.L., 28911 Leganés, Spain
[3]National Inter-University Consortium for Telecommunications (CNIT), 56124 Pisa, Italy
[4]IMDEA Networks Institute, 28918 Leganés, Spain
[5]Capgemini Engineering, 28022 Madrid, Spain
[6]Faculty of Science, University of Amsterdam, 1012 WX Amsterdam, The Netherlands
[7]Ericsson Research, 56124 Pisa, Italy
[8]Institute of Communication, Information and Perception Technologies, Scuola Superiore Sant'Anna, 56127 Pisa, Italy
[9]Institute of Electronics, Computers and Telecommunication Engineering, National Research Council (IEIIT-CNR), 10129 Turin, Italy
[10]MIRANTIS, 61166 Kharkiv, Ukraine

Corresponding author: Chia-Yu Chang (chia-yu.chang@nokia-bell-labs.com)

This work was supported in part by the EU H2020 5GROWTH Project under Grant 856709.

ABSTRACT Network slicing plays a key role in the 5G ecosystem for verticals to introduce new use cases in the industrial sector, i.e., Industry 4.0. However, a widely recognized challenge of network slicing is to provide traffic isolation and concurrently satisfy diverse performance requirements, e.g., bandwidth and latency. Such challenge becomes even more important when serving a large number of network traffic flows under a resource-limited condition between distributed sites, e.g., factory floor and remote office. In this work, we present the capability to retain these two goals at the same time, by applying the virtual queue notion over a priority queuing based pipeline in P4 switch over software-defined networks. To examine the effectiveness of our approach, a proof-of-concept is setup to serve different requests of Industry 4.0 use cases over a mixed data path, including P4 switch and Open vSwitch, for a large number of network flows.

INDEX TERMS Communication system traffic control, performance isolation, network slicing, OpenFlow, P4, quality of service, software defined networking.

I. INTRODUCTION

The Fifth Generation (5G) of mobile networks reshuffles the communication industry ecosystem, carving out a path to providing ubiquitous connectivity and value-added services to vertical industries. Therefore, a variety of new use cases have emerged under the umbrella of 5G with diverse service requirements, ranging from substantial bandwidth (e.g., by high definition streaming) to stringent low-latency (e.g., by digital twin systems). To this end, 5G is designed to provide vertical industries with networking environments tailored to their needs, sharing the same physical infrastructure, that is, network slicing.

The associate editor coordinating the review of this manuscript and approving it for publication was Petros Nicopolitidis.

Several standards development organizations have elaborated on definitions to capture the notion of a network slice. To cite a few, Next Generation Mobile Networks (NGMN) alliance defines a Network Slice Instance (NSI) as "…a set of network functions, and resources to run these network functions, forming a complete instantiated logical network to meet certain network characteristics required by the service instance(s) …" [1]. Third Generation Partnership Project (3GPP) defines a NSI in [2] as "…a set of network function instances and the required resources (e.g., computing, storage, and networking resources) that form a deployed network slice". Moreover, European Telecommunications Standards Institute (ETSI) NFV EVE012 [3] articulates the correspondence between a network slice from 3GPP and a network service from ETSI NFV. There, it describes that the NFV Network Service (NFV-NS) can be regarded as a resource-centric

view of a network slice, for the cases where a NSI would contain at least one virtualized network function. In short, a network slice can be viewed as a virtual network overlaid on the shared physical network.

Going a further step, a key challenge to accommodate several network slices over the same physical resources, as stated in [4], is to serve their different needs based on the individual service type they carry. A common requirement to this end, is slice *connectivity isolation*, i.e., users of a slice cannot communicate with users/services residing in other slices. For example, in a layer 2 network, this requirement can be satisfied by enforcing traffic separation through VLAN. Another demanding requirement for network slicing is *performance isolation* [5], where the resources utilized by a slice should not negatively affect the performance of another slice. Such resources can span CPUs, storage space, and network capacity, whereas performance parameters, such as bandwidth and delay guarantees, can vary.

Depending on the underlying network characteristics, some solutions naturally retain performance isolation. For example, in an optical wavelength division multiplexed network, an isolated slice with guaranteed bandwidth can be achieved simply by preserving a wavelength for it. It is worth mentioning that such a guaranteed bandwidth is not trivial in a traditional layer 2 network using Ethernet switches. In fact, performance isolation can be achieved by exploiting network programmability enabled by Software-Defined Networking (SDN). SDN provides both a standard protocol (e.g., OpenFlow) to program network devices and a standardized vision of network devices (e.g., a set of flow tables that can be programmed with a number of flow rules) [6]. Nevertheless, OpenFlow-enabled switches exhibit fixed behavior, as specified in the datasheet of the switch ASIC. More recently, (re)programming the functionality of forwarding devices has been enabled using domain-specific language, such as P4 [7]. Thus, P4 is designed to be platform-independent and can offer a generic data-plane programmability method.

In addition, one complementary challenge is to manage and control an End-to-End (E2E) network slice over the underlying infrastructure, toward the verticals. Such operations must be performed over heterogeneous programmable data-planes, supporting different levels of programmability, and expose them via a unified interface. Take Open vSwitch (OvS) and P4 reference software switch (bmv2) as examples, both support performance isolation, to a certain extent. However, corresponding control-plane applications should be aware of their different programmability capabilities to support the requested performance metrics.

To answer the challenges raised above, this article proposes a solution guaranteeing performance isolation for multiple network slices, sharing a single packet-based, software-defined transport network. Our solution enables a vertical-specific networking environment over a data-plane comprising both OpenFlow and P4-enabled software switches, by (1) extending the 5Growth system architecture [8] to model vertical's requirements into QoS policies, (2) enhancing QoS control application over SDN controller toward mixed switches in the data-path, and (3) applying the virtual queue solution for three performance isolated Industry 4.0 use case. To the best of our knowledge, this is the first time a solution is presented that enables performance isolation for network slicing seamlessly over a transport network composed of networking devices with different levels of programmability, using a 5G service platform and SDN controller.

The remainder of this paper is organized as follows. A brief state-of-the-art review is provided in Sec. II, including current data-plane solutions on performance isolation and the 5Growth baseline architecture. We elaborate on our solution in Sec. III, in which two separate aspects are covered: (i) 5Growth architecture enhancements and (ii) SDN network extensions in terms of control-plane applications and data-plane programmability to support network slice performance isolation. In addition, we validate our solution over the Proof-of-Concept (PoC) presented in Sec. IV in terms of serving different service requests for particular use cases, drawn from the 5Growth Industry 4.0 pilot, over hundreds of network traffic flows. Finally, concluding remarks and future research directions are presented in Sec. V.

II. RELATED WORK AND BACKGROUND

A. DATA-PLANE PERFORMANCE ISOLATION

Performance isolation in SDN-based network slicing is a relatively recent research topic. Specifically, several related studies have used SDN in conjunction with different technologies to implement traffic metering and traffic prioritization, which are key features behind performance isolation. These prior arts can be classified according to their applied approach: OpenFlow [6] or P4 [9].

1) OPENFLOW APPROACH

The works in [10], [11] use OpenFlow meters and transmission queues to provide bandwidth guarantees; however, the proposed solutions are not integrated with the concepts of network slicing. More in particular, [10] utilizes a multi-table pipeline, where the first table is used to direct the packets toward the meters for distinguishing the traffic in three Quality of Service (QoS) categories, and the second table is used to direct the three traffic categories toward different transmission queues. Conversely, [11] proposes and implements an extension of OpenFlow meters together with a meter rate evaluator mechanism to support a hierarchy of meters. Exploiting the defined hierarchy, the threshold of a meter is dynamically determined at the device level, based on actual traffic measured by higher priority meters.

Some further works extend the notion of bandwidth guarantee along different directions. In [12], Multi Protocol Label Switched (MPLS) tunnels are utilized to provide an E2E bandwidth guarantee, relying on both OpenFlow meters and QoS information in the MPLS header of each packet. By utilizing the OpenFlow meter together with traffic prioritization,

the work in [13] can guarantee the required bandwidth for emergency flows and maximize the best-effort flows over the remaining bandwidth.

Additionally, the authors of [5] apply a solution similar to that in [10] but support a framework for deploying a set of network slices in an SDN network using OpenFlow protocol. Specifically, an Open Network Operating System (ONOS) SDN controller is used to create and deploy network slices providing both connectivity and performance isolation in terms of the guaranteed bandwidth. The deployed solution not only utilizes the Virtual Private LAN Service (VPLS) application to create the connectivity-isolated multipoint-to-multipoint network slices, but also introduces the QoSlicing application to enforce the bandwidth threshold for each slice using OpenFlow meters.

2) P4 APPROACH

The P4 specification provides a standard method for bandwidth management using packet classification and metering in [9], in which packet classification uses "two rate Three Color Marker" (trTCM) scheme standardized by Internet Engineering Task Force (IETF) in [14] and stateful meter object is used to record statistics of a flow. This information can be utilized to perform different actions, i.e., mark or drop packets, according to the burstiness and bit rate criteria. By extending the basic P4 meter, the authors of [15] introduce a multi-color marker to support priority differentiation for each isolated virtual network with dedicated bandwidth guarantee. Compared with OpenFlow, it has been argued in [16] that P4 meters are more flexible in terms of band type (i.e., associated operation); however, P4 meters are statically defined by the P4 program with programmable trTCM parameters. Note that meters are only exposed to P4 programmers as an external function that can be used to call metering built-in functionality, which may not be available for all P4 targets, as in the case of NetFPGA-SUME [17], or may have proprietary or limited behavior, as in the case of Netronome SmartNICs that use a single rate/burst pair of limits and thus can report only two states for a packet.

To address such issues, the authors of [18] introduce the concept of virtual queues in the P4 pipeline, enabling traffic management for different P4 programmable targets. A virtual queue is a mechanism used to model the sojourn delay of packets in the queue, as if the packets arriving in the real queue were served by a link with a bandwidth lower than the actual capacity of the link. This virtual delay can be used to enforce traffic policing or even drive Active Queue Management (AQM). It is worth noting that such virtual queue mechanism is implemented using commonly supported P4 constructs (i.e., registers), such that it can be easily ported to different P4 targets with similar performance (in terms of processing delay), such as Netronome SmartNICs [18]. Based on this concept, our prior work [19] allows the setting of distinct bandwidth and delay guarantees for each performance-isolated slice, by associating a separate virtual queue for each slice.

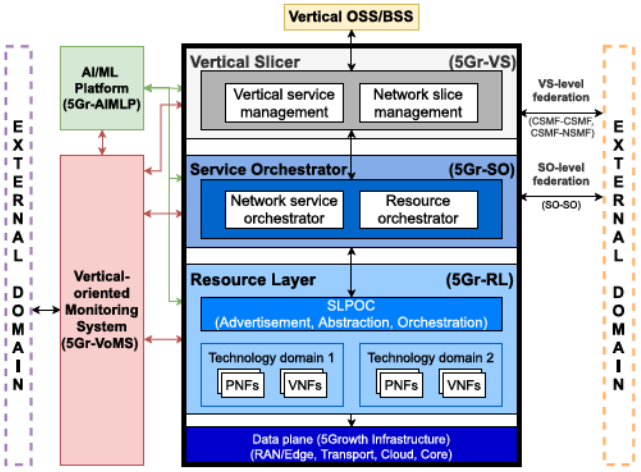


FIGURE 1. 5Growth high-level architecture.

3) DISCUSSIONS

Based on the above recaps according to the applied approach, in this work we consider a mixed data path for the Industry 4.0 use case. Indeed, while the P4 language enables programmable packet processing in a protocol-independent manner, the current P4 software switch has a more limited performance bottleneck than the Open vSwitch.[1] Thus, in 5G testbeds (e.g., 5TONIC [20] and Fed4FIRE+ [21]), a clean-slate replacement of all Open vSwitch switches with P4 software ones would request thorough performance evaluation. A mixed data-path can take advantage of both worlds. In summary, replacing some OpenFlow software switches with P4 enabled ones, endows the network with programmability features while keeping high-performance thanks to the presence of the Open vSwitch switches.

B. 5GROWTH ARCHITECTURE: AN OVERVIEW

To exploit data-plane performance isolation solutions for vertical needs, the 5Growth architecture is applied and evolved from its baseline view in [8]. Such evolution aims to build a vertical-oriented platform through a set of innovations, e.g., smart orchestration and resource control, closed loop operations driven by Artificial Intelligence and Machine Learning (AI/ML), to retain usability, flexibility, automation, performance and security for new vertical use cases.

A sketch of the 5Growth architecture is presented in Fig. 1; it is composed of three core building blocks, namely Vertical Slicer (5Gr-VS), Service Orchestrator (5Gr-SO) and Resource Layer (5Gr-RL), complemented by the Vertical-oriented Monitoring System (5Gr-VoMS) and the AI/ML Platform (5Gr-AIMLP). It is worth noting that the source codes of these building blocks are open and can be found in their respective GitHub repositories [22]–[26]. In the following, we briefly outline these functionalities.

The **5Gr-VS** acts as a single entry point for verticals to request customized network services, through its

[1]It is worth mentioning that some works (e.g., https://github.com/osinstom/P4-OvS) aims to develop high-performance P4 software switch.

NorthBound Interface (NBI), oriented toward the vertical Operations/Business Support System (OSS/BSS). In particular, service requests are submitted via such an interface, and afterwards, the 5Gr-VS will map the requested vertical services to network slices via translation and arbitration and finally forward requests to the 5Gr-SO. In addition, such block is responsible for lifecycle management of both vertical services as well as network slice instances.

The **5Gr-SO** provides both network service and resource orchestration capabilities to support the E2E orchestration of NFV-NSs and their lifecycle management. The NFV-NSs can be provisioned across one or more administrative domains depending on the service requirements and the availability of network services/resources offered by the local and peering domains. Furthermore, the 5Gr-SO is responsible for handling the Service Level Agreement (SLA) compliance for a given NFV-NS and triggering the scaling process as a reaction to SLA violation. To this end, the 5Gr-SO offers an integrated view of the E2E network service to the 5Gr-VS.

The **5Gr-RL** bridges the gap between NFV-NS view from 5Gr-SO and domain-specific physical/virtual network functions (PNFs/VNFs), as well as resources (e.g., computing, networking, and storage). Therefore, its two main components are the Abstraction Engine (AE) and Resource Orchestrator (RO). The former provides an abstracted view of the available resources tailored to the 5Gr-SO, whereas the latter orchestrates resources and instantiate VNFs over the underlying infrastructure. With multiple technology domains under its control, the 5Gr-RL manages the underlying infrastructures through different plugins as a Single Logic Point of Contact (SLPOC), providing a common abstraction view of the managed resources to the 5Gr-SO via the IFA005-based interface [27].

The **5Gr-VoMS** provides a platform for collecting, storing, processing, analyzing, and visualizing data information, based on the metrics requested by the deployed services and vertical-application monitoring probes. It utilizes Messaging Queues (MQs) to exchange messages between the monitoring platform and vertical-side monitoring agents, in order to configure them and receive feedback data on demand. Such dynamic monitoring can support innovative mechanisms related to reliability (via self-healing and auto-scaling), control-loop stability, and analytical features (such as forecasting and anomaly detection).

The **5Gr-AIMLP** is designed to support different 5Growth building blocks (i.e., 5Gr-VS, 5Gr-SO, 5Gr-RL) to execute AI/ML algorithms for their decision-making processes, e.g., NFV-VS scaling closed-loop at 5Gr-SO, which might require training from external data. For each decision to be made, the corresponding 5Growth building block will select the model among those offered by the 5Gr-AIMLP catalog. Depending on the nature of the model, it will be trained offline or online through data captured by the 5Gr-VoMS, which will provide the metrics of interest.

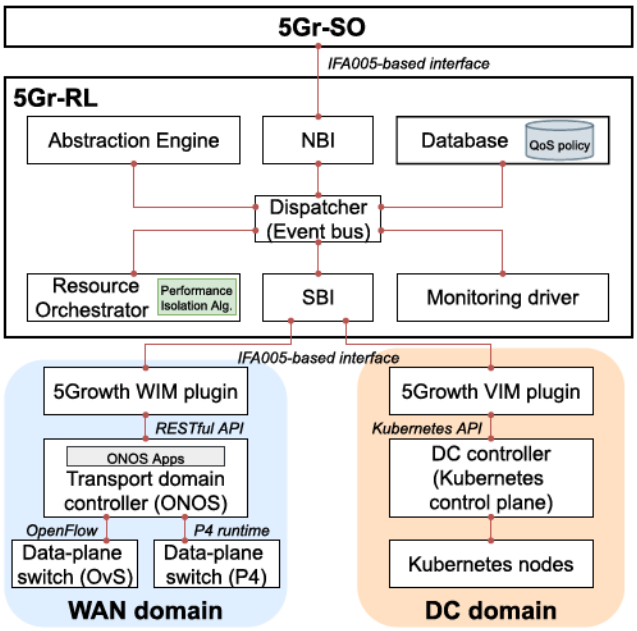


FIGURE 2. 5Gr-RL architecture and plugins.

III. PERFORMANCE ISOLATION IN 5GROWTH

This section presents our solution for guaranteeing performance isolation for multiple network slices and how it is integrated into the 5Growth platform. First, we discuss the extensions to the 5Gr-RL component and their implementation in the form of plugins (Sec. III-A). In the following, we elaborate on our innovations on both control- and data-plane to support bandwidth and delay guarantees (Sec. III-B).

A. EXTENSIONS ON 5Gr-RL AND PLUGINS

1) 5GROWTH RESOURCE LAYER EXTENSIONS

The 5Gr-RL acts as the NFV Infrastructure (NFVI) manager in the 5Growth architecture and is responsible for providing physical resources (e.g., computing and networking) to VNFs. The architecture is depicted in Fig. 2. The IFA005-based interface is exposed by the 5Gr-RL toward both 5Gr-SO (i.e., NBI) and VIM/WIM plugins (i.e., SouthBound Interface [SBI]) to receive and send commands, respectively. As mentioned in Sec. II-B, the two main building blocks of 5Gr-RL are AE and RO. The former gathers information about the managed resources from the underlying plugins and provides an E2E abstracted view of resources to the 5Gr-SO; the latter is responsible for orchestrating resource provisioning to accommodate service requests.

In this work, we extended the RO with a *performance isolation algorithm* integrated into the overall resource orchestration process, as shown in step 4 of Fig. 3. This algorithm receives a list of available QoS policies from 5Gr-RL database (step 0 in Fig. 3), where each policy expresses the capabilities of the underlying infrastructure(s) to retain performance isolation. Each QoS policy contains:

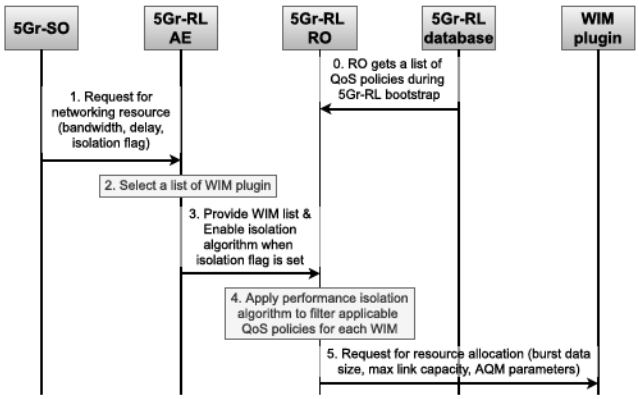


FIGURE 3. 5Gr-RL resource orchestration steps.

i) A preference ID, identifying the QoS policy
ii) A list of WIM plugin identifiers, defining to which plugin this policy is feasible
iii) Properties supported by the QoS profile
iv) Minimum and maximum bandwidth supported by the QoS profile
v) Minimum delay supported by the QoS profile
vi) Burst size of data packets
vii) Maximum capacity of the link
viii) AQM parameters

By utilizing different QoS policies, the overall resource orchestration process can address the QoS requirements of the 5Gr-SO. In more detail, when a request for networking resources arrives via NBI, the AE will first compose a list of WIM plugins that will provide resources according to the bandwidth and delay requirements. Subsequently, if the slice isolation flag is set at the request, identifying that slice performance isolation is necessary for networking resources, the *performance isolation algorithm* in Alg. 1 at the RO will be invoked with two inputs: A list of WIM plugins with individual QoS policies (*QoS_Pol* in Alg. 1) and the QoS request (*QoS_Req* in Alg. 1).

In detail, for each WIM plugin in the list, the algorithm will filter out all inapplicable QoS policies according to the following criteria: a QoS policy is considered "applicable" if it contains the requested bandwidth within its minimum/maximum bandwidth range and it provides a lower or equal minimum delay with respect to the requested one. Note that if more than one QoS policy passes the filtering process above, the policy with the lowest preference identifier will be selected (*SelectID* in Alg. 1). Finally, the RO will send resource allocation requests (*RA_Req* in Alg. 1) to the corresponding WIM plugin via the SBI, comprising parameters such as data burst size, maximum link capacity, and AQM parameters, to apply the QoS policy to the SDN controller and the underlying data-plane infrastructures. It is worth noting that an operation exception event will be sent back to the 5Gr-SO, including the given operation and message, once no QoS policy is chosen. Hence, the orchestration procedure shown in Fig. 3 will be viewed as failed, and the re-orchestration procedure will modify its request until it succeeds. A more efficient solution is to make 5Gr-SO provide a list of alternative QoS requests in which some relaxed resource requests (e.g., a requirement with lower bandwidth and/or higher delay) are provided to avoid orchestration failure.

Algorithm 1: Performance Isolation Algorithm

```
Input  : QoS_Pol is the list of WIM plugin with QoS policy
         QoS_Req is the networking resource requirement
Output: RA_Req is the resource allocation request
begin
    SelectID ← ∞;
    ReqBW ← QoS_Req.get_BW();
    ReqDelay ← QoS_Req.get_Delay();
    for policy ← 1 to number of QoSPolicy do
        MaxBW ← QoS_Policy.get_Max_BW(policy);
        MinBW ← QoS_Policy.get_Min_BW(policy);
        MinDelay ← QoS_Policy.get_Delay(policy);
        /* Filter based on requested bandwidth and delay*/
        if MinBW ≤ ReqBW ∧ MaxBW ≥ ReqBW then
            if MinDelay ≤ ReqDelay then
                /* QoS policy is consider applicable*/
                if policy ≤ SelectID then
                    SelectID ← policy
    if SelectID ≠ ∞ then
        RA_Req.burst_size ← QoS_Pol.get_Burst(SelectID);
        RA_Req.link_cap ← QoS_Pol.get_Link(SelectID);
        RA_Req.aqm_para ← QoS_Pol.get_AQM(SelectID);
    else
        Reply operation exception event to the 5Gr-SO;
```

2) WAN INFRASTRUCTURE MANAGER PLUGIN

The WIM plugin is a dedicated component that enables the interaction between the 5Gr-RL and the ONOS SDN controller. It exposes two interfaces: an IFA005-based NBI to communicate with the above 5Gr-RL, and a RESTful SBI to interact with the underlying SDN controller. Such a plugin allows the 5Gr-RL to (i) query the current status of networking resources, and (ii) request the allocation/release of resources from the managed infrastructure.

On the one hand, upon receiving the query request from the 5Gr-RL, the WIM plugin provides an "abstracted" view to the 5Gr-RL by representing the transport network as a set of *logical links* interconnecting NFVI Point-of-Presence (NFVI-PoP) gateways. Such logical link abstraction is technology-agnostic (i.e., independent from the fact that the physical link is an optical or an MPLS link), and it reveals only performance-related information (e.g., delay, available bandwidth, total bandwidth) to the 5Gr-RL. Hence, any details regarding the WAN domain (e.g., switch characteristics, switch inter-connectivity, traffic steering features) are not disclosed.

On the other hand, the WIM plugin will receive the resource allocation request for a specific logical link between two NFVI-PoP gateways. Within this request, several identifiers and parameters are included. The first is *logicalLinkId* together with both *srcGwIPAddress* and *dstGwIPAddress* parameters to uniquely identify a single logical link and assign the IP addresses of the NFVI-PoP gateway

endpoints (according to the data flow direction). Also, specific QoS parameters (cf. Fig. 3) are included in the request to be enforced. After receiving this request, the WIM plugin will first check the resource availability and send another request to the ONOS SDN controller to (1) set up a forwarding path between the specified NFVI-PoP gateway IP addresses and (2) enforce QoS parameters over the intermediate data-plane infrastructures. The SBI is implemented as a RESTful HTTP interface. Two POST requests are used for slice performance isolation: one to set up the slice, and another to apply the selected QoS policy.

3) VIRTUALIZED INFRASTRUCTURE MANAGER PLUGIN

The VIM plugin manages computing and communication resources for a specific VIM domain. Its goal is to translate the requests from the 5Gr-RL via the IFA005-based NBI to the Kubernetes API in order to create/delete pods or to get pod status inside Kubernetes cluster. Nevertheless, the initial VIM plugin for Kubernetes can only apply a single default configuration without supporting the VLAN-based external connection to the pods. Thanks to the introduction of Container Network Interface (CNI) [28], multiple new network technologies can be added to extend Kubernetes capabilities. Therefore, we adopt the Multus CNI [29] for Kubernetes and extend the VIM plugin for managing the Multus CNI in order to (i) attach multiple network interfaces to the pods, and (ii) connect pods to external VLAN-based transport network managed by the WIM plugin.

B. PERFORMANCE ISOLATION IN SDN NETWORKS

1) CONTROL-PLANE SDN APPLICATIONS

To deploy isolated network slices, the ONOS SDN controller is used in the WAN domain, as depicted in Fig. 2. Specifically, the standard ONOS distribution includes the VPLS application to create L2 multipoint-to-multipoint connections over the shared transport network for a network slice. Nevertheless, to make it compatible with the SBI of the WIM plugin (cf. Fig. 2), we have implemented a REST API for the VPLS application. The new interface has been submitted to the ONOS community and merged into release version 2.4.0.[2] It is utilized by 5Gr-RL to issue commands through its SBI and exploit the connectivity isolation provided by the VPLS application.

To support performance isolation among network slices created with the VPLS application, we rely on the *QoSlicing* application introduced in our previous work [5], which was designed to support and control meters within OpenFlow-based switches. Fig. 4 illustrates all building blocks of this application including its interactions with the VPLS application, the WIM plugin, and the ONOS core. The slice profile manager within the *QoSlicing* application imports the currently deployed slices' information from the VPLS application and stores it in a local data-store (i.e., the slice profile store). Moreover, via the interface toward the WIM plugin in its north bound, the slice profile manager receives slice-specific QoS parameters (cf. Fig. 3) to be enforced in the data-plane. Finally, the QoS profile of each slice is created and forwarded to the operation handler to be translated into a set of OpenFlow meter and flow rules. These rules will be deployed on the data-plane via the northbound APIs of the ONOS core.

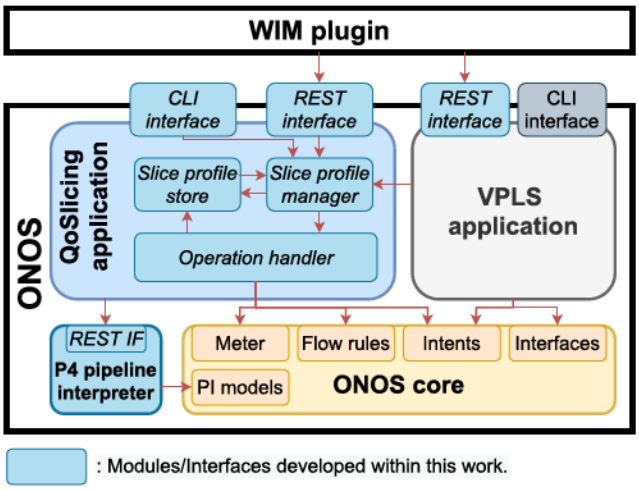


FIGURE 4. Applied ONOS SDN controller with applications.

On top of the above vanilla functionalities, the *QoSlicing* application has been enhanced in this work[3] in three aspects: (1) adding support for P4-based switches to enable the management of network slices along a mixed datapath, (2) configuring the parameters for a slice-specific virtual queue in the P4-based switch, and (3) introducing the REST API to interact with different applications and the WIM plugin. First, to support P4-based switches, we comply with the P4runtime specification[4] and use the gRPC protocol for the interaction between the *QoSlicing* application and P4-enabled switches. In detail, the gRPC client is employed in *QoSlicing* applications and the gRPC server is placed in the P4-based bmv2 software switch, and a mapping function is introduced to translate the OpenFlow "meterId" format into an identifier supported by P4 specification. Second, the *QoSlicing* application has been extended to support the configuration of slice-specific virtual queues in the P4 pipeline for both delay and throughput guarantees. In specific, these configured parameters will be placed in the egress table and take effect in the egress pipeline (will be introduced later in Sec. III-B2). Finally, we extend the *QoSlicing* application to support REST API toward VPLS application, P4 pipeline interpreter application, and WIM plugin to flexibly push the corresponding QoS parameters from the management plane, beyond the legacy command-line interface, through the SBI of 5Gr-RL.

In addition to the above applications, we added support for ONOS to control P4 pipeline. The initial design of ONOS

[2]https://wiki.onosproject.org/display/ONOS/Downloads

[3]https://github.com/5growth/5gr-rl/tree/master/i8-code/QoS-Slicing

[4]https://github.com/p4lang/p4runtime

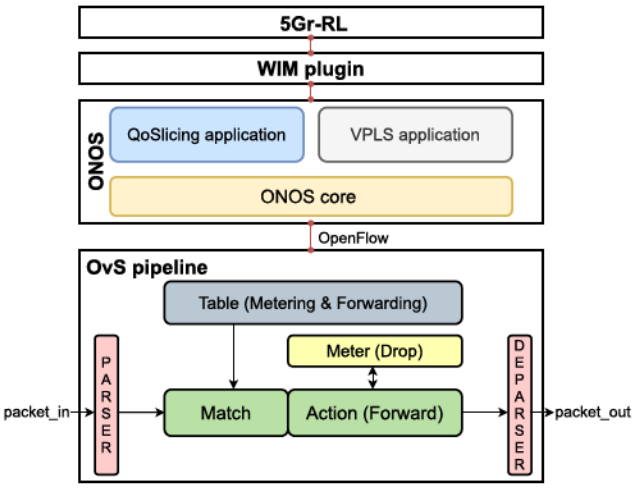


FIGURE 5. Applied OvS processing pipeline.

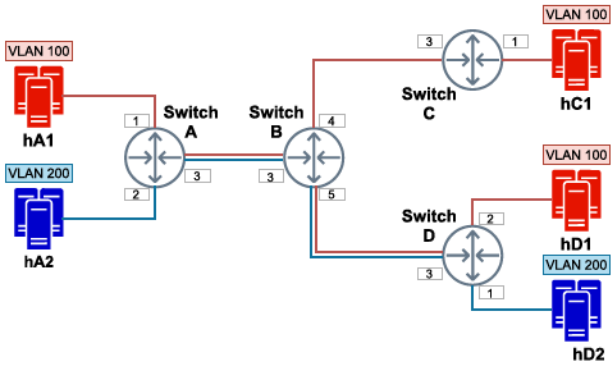
with regard to the management of the OpenFlow protocol is based on an immutable pipeline. Therefore, an interpreter is required to control and configure P4 switches with mutable pipeline. The new interpreter exposes methods that allow ONOS to communicate with the underlying P4 program and build an independent pipeline. In particular, it leverages PI models provided by the ONOS core (cf. Fig. 4) and applies a mapping between each ONOS and the corresponding P4 actions, which are derived from the P4 program. Moreover, this interpreter exposes a REST API toward the *QoSlicing* application to obtain QoS parameters, e.g., AQM parameters for virtual queues in P4-bas switches.

2) PROGRAMMABLE DATA-PLANE TRAFFIC MANAGEMENT

To support performance isolation in the transport network and fulfill the requirements of each slice, the ability to (re)program the data-plane infrastructure is essential. We introduce the data-plane pipelines for both OvS and P4 bmv2 switches, controlled by ONOS using the aforementioned applications.

The employed OvS processing pipeline is shown in Fig. 5, as well as the two ONOS applications controlling it, i.e., *QoSlicing* and VPLS. The table in the pipeline determines the metering and forwarding operations of the incoming packets. In particular, the relevant fields of the incoming packet are compared and matches against table entries and then specific actions takes place. Moreover, the OpenFlow meter is used for traffic policing.

To depict how the metering and forwarding operations work, an example scenario is sketched in Fig. 6a, with two slices over a simple network topology. Note that the red slice has VLAN id 100 and is configured over three hosts (i.e., hA1, hC1, hD1), whereas the blue slide has VLAN id 200 and is configured over two hosts (i.e., hA2, hD2). The table for switch B is shown in Fig. 6b. The entries are installed by the VPLS application and updated by the *QoSlicing* application, resulting in a dedicated entry for each combination of input port and destination host, i.e., six entries for VLAN



(a) Example topology consisting of two slices, four switches, and six hosts

TABLE ID	FLOW MATCH	INSTRUCTIONS
1	VLAN 100, in_port 3, dst_mac: hC1	APPLY_ACTIONS(output port 4, meter104)
2	VLAN 100, in_port 3, dst_mac: hD1	APPLY_ACTIONS(output port 5, meter105)
3	VLAN 100, in_port 4, dst_mac: hA1	APPLY_ACTIONS(output port 3, meter103)
4	VLAN 100, in_port 4, dst_mac: hD1	APPLY_ACTIONS(output port 5, meter105)
5	VLAN 100, in_port 5, dst_mac: hA1	APPLY_ACTIONS(output port 3, meter103)
6	VLAN 100, in_port 5, dst_mac: hC1	APPLY_ACTIONS(output port 4, meter104)
7	VLAN 200, in_port 3, dst_mac: hD2	APPLY_ACTIONS(output port 5, meter205)
8	VLAN 200, in_port 5, dst_mac: hA2	APPLY_ACTIONS(output port 3, meter203)

METER ID	METER BANDS
103	(Type: DROP; Rate: 20Mb/s; Burst: 10Mbyte)
104	(Type: DROP; Rate: 20Mb/s; Burst: 10Mbyte)
105	(Type: DROP; Rate: 20Mb/s; Burst: 10Mbyte)
201	(Type: DROP; Rate: 30Mb/s; Burst: 15Mbyte)
203	(Type: DROP; Rate: 30Mb/s; Burst: 15Mbyte)

(b) Table for switch B programmed by the VPLS and QoSlicing applications

FIGURE 6. An example scenario with two established slices and the programmed table for metering and forwarding.

id 100 and two entries for VLAN id 200. For each entry, the "flow match" column specifies the fields to be matched while the "instruction" field states the corresponding actions to be taken. For instance, an incoming packet to switch B port 5 with VLAN id 100 and destination hC1 MAC address matches the 6-th entry of the table; therefore, it is metered using meter 104 and forwarded to port 4 (if not dropped). The corresponding meter measures the traffic flow and drops the packet if the rate limit is exceeded, with some tolerance depending on the configured maximum burst size.

In contrast, the bmv2 software switch provides P4-programmable packet processing for a customized data-plane pipeline. P4 is a high-level domain-specific programming language used to define customized packet processing, including parsing, matching actions at ingress/egress pipelines, and deparsing. A P4 program must be compiled by the P4 Compiler (P4C) in order to load it into a switch. The compilation produces two distinct outputs: (i) Target-specific binaries for pipeline implementation, and (ii) Target-independent P4Info file for pipeline runtime control. Fig. 7 depicts the applied P4 processing pipeline. The ONOS *QoSlicing* application controls the pipeline using the P4runtime API over the aforementioned gRPC protocol.

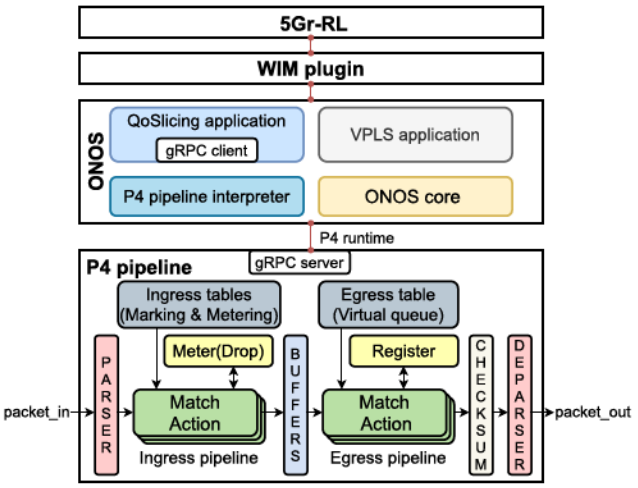


FIGURE 7. Applied P4 processing pipeline.

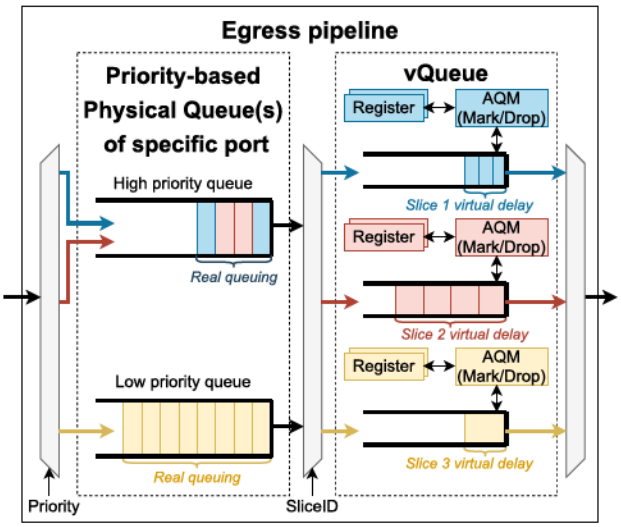


FIGURE 8. Virtual queues at P4 egress pipeline.

When an incoming packet enters the pipeline, its header is parsed in order to obtain its Ethertype and VLAN tag values. Afterwards, at the ingress pipeline, two tables are used for two distinct types of actions:

i) Packet marking actions based on the parsed VLAN tag value to assign the output port, the slice identifier (i.e., sliceId), and priority level to be used for further actions at the egress pipeline, and
ii) Meter "coloring" actions that use the indexed meters to tag a packet for dropping due to the rate limitation.

Alternatively to the use of meters for traffic policing, virtual queues can be applied at the P4 egress pipeline if delay guarantees also have to be met, by introducing an additional table in Fig. 7. In detail, Fig. 8 provides a graphical visualization of the virtual and physical queues at the egress pipeline. In concept, each slice has a dedicated virtual queue, and every egress packet (from a physical queue) will virtually spend a sojourn time in this queue as if it waits after the previous egress packet of the same slice. The sojourn time in this virtual queue is computed by multiplying the egress packet size in bytes by the slice-specific elementary delay in seconds per byte. Therefore, based on the previously allocated sliceId, each egress packet will be put into the slice-specific virtual queue, and by using the read/write operations of register in P4, the sojourn time in the virtual queue can be accumulated from packet to packet as the virtual delay.

Consequently, the AQM mechanism will be applied to either drop or mark packets, based on such slice-specific virtual delay as well as the non-slice-specific physical delay that is spent in the physical queue, to slice-specifically limit the throughput and control the experienced delay. As shown in Fig. 8, multiple physical queues can be used to differentiate the real queuing delay experienced by different priority levels by applying strict non-preemptive priority queuing and assigned priority levels, supported by the bmv2 non-programmable traffic manager. It is worth mentioning that the additional physical queues will require extra computing resources, and because of the available L2 Class of Service (CoS) values, its number usually cannot be configured to more than eight. Finally, the checksum fields are updated, and the deparsing operations occur at the end of the pipeline, as shown in Fig. 7.

IV. PROOF-OF-CONCEPT PERFORMANCE RESULTS

To show the effectiveness of our work for the industry sector, in this section we first elaborate the applied multi-service traffic profile based on the use cases from both 5G Alliance for Connected Industries and Automation (5G-ACIA) and a real-world Industry 4.0 pilot. Finally, we present and discuss the measured performance of our PoC in detail.

A. APPLIED INDUSTRY 4.0 USE CASES

Industrial applications have specific performance requirements; thus, it is necessary to analyze the data volume and traffic characteristic of each selected use case scenario before deployment. In this sense, 3GPP collects several use cases for the future factory in [30], and 5G-ACIA provides the traffic models in [31] that focus on the use case categories for industrial automation, process automation, and inbound logistics. In particular, seven traffic models are introduced for 5G-ACIA use cases: (1) Motion control, (2) Closed-loop control, (3) Process monitoring, (4) Mobile robot, (5) Human-Machine Interface (HMI), (6) Closed-loop control for process automation, and (7) Control-to-control. Each traffic model contains a set of parameters used for the corresponding application function that participates in data communication over the network. Note that 5G-ACIA also provides the aggregation level of each use case for factory layout deployment.

Under the umbrella of the 5Growth project [32], three specific industry sectors (Industry 4.0, energy, and transportation) are targeted for new technology roll-outs over different vertical pilots. Among these pilots, we aim to validate our network slicing approach based on the three use cases identified for the Comau Industry 4.0 pilot [33]. The first *digital twin* use case aims to (i) remotely control and (ii) virtually reproduce the Comau production line in real-time.

TABLE 1. Mapping between 5Growth Comau industry 4.0 pilot and 5G-ACIA use cases.

Slice ID	Comau pilot use case	Mapped 5G-ACIA use case(s)	Traffic characteristic	Latency requirement	Number of flows	Peak aggregated data rate
1	Digital twin	Motion control, Closed-loop control	CBRs (Remote sensors, Remote actuator)	15 ms	150	171.36 Mbps
2	Telemetry	Process monitoring	CBRs (Remote sensors)	30 ms	35	1.90 Mbps
3	Remote support	Mobile robot, HMI	VBR (HD video streaming), CBR (Remote controller)	50 ms	15	128.16 Mbps (CBR/VBR: 7.54/120.62 Mbps)

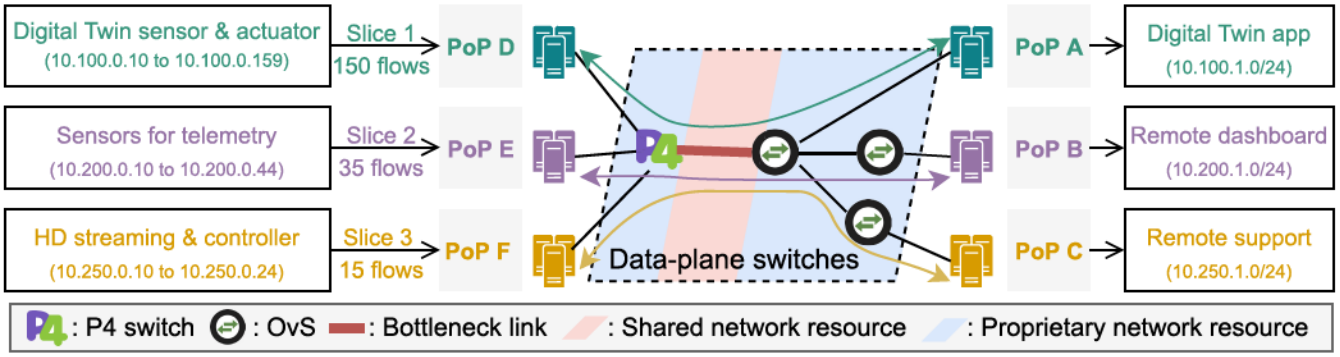


FIGURE 9. Data-plane switches of proof-of-concept.

The second *telemetry* use case enables system monitoring and failure prevention through data collection from sensors (e.g., vibration, pressure, temperature) at the plant and equipment. The last is *remote support*; it provides on-site assistance from skilled technicians located at remote sites by using on-demand high-definition video streaming.

In TABLE 1 we map the Comau pilot to the 5G-ACIA use cases, and identify each one to be used in the PoC with an individual slice ID. Note that these three slices have different traffic characteristics (i.e., Constant Bit Rate [CBR] or Variable Bit Rate [VBR]) for each flow and distinct latency requirements for the respective application. Finally, based on the model parameters from the mapped 5G-ACIA use case, the number of flows is provided in the same table and the peak aggregated data rate is calculated.

B. PROOF-OF-CONCEPT: SETUP AND PERFORMANCE

To validate the performance of the three network slices in TABLE 1, a PoC is set up in a lab environment to provide a realistic scenario that can transport all 200 traffic flows between slice-specific pairs of Point of Presences (PoPs) in a data-plane containing both OvS and P4 bmv2 switches, as shown in Fig. 9. The data-plane includes both proprietary (owned by vertical) and shared network resources (offered by Communication Service Provider [CSP]) to represent the integration of public/non-public network in Industry 4.0 [34].

In detail, we assume one bottleneck link in the data-plane, providing the leased resource from the CSP on the shared medium, such as an overlay passive optical network or wireless spectrum. A common approach would be to over-provision its capacity for all three slices according to the peak aggregated data rate, e.g., 300 Mbps. Nevertheless, such capacity over-provisioning cannot always be met due to the introduction of 5G radio access for Industry 4.0 use case, where the spectral efficiency (in bps/Hertz) varies depending not only on the spectrum bandwidth but also on the radio signal strength. In practice, sometimes the capacity will be lower than its peak rate owing to user mobility or wireless channel volatility. In that sense, some services (e.g., video streaming) may have their respective self-adaptation schemes and can adjust their VBR flow throughput to maintain a certain level of Quality of Experience (QoE) [35], e.g., applying Dynamic Adaptive Streaming over HTTP (DASH) streaming. Nevertheless, not every service is adaptive, and this challenging condition needs to be seriously addressed by CSP. To this end, we selected a bottleneck of 60% (180 Mbps) of the aggregated input data rate because it well represents the situation of congestion in the network, which is sufficient for the experiments conducted in this paper. In addition, investigating the behavior of our application with different bottleneck values is an interesting future direction for this work.

On top of the aforementioned data-plane switches, our PoC setup can be found in Fig. 10 within one server equipped with 8-core Intel Xeon Gold 6244 CPUs @3.60GHz and 64GB RAM. All switches are deployed using mininet[5] and controlled by the ONOS SDN controller. Also, the 5Gr-RL is deployed to orchestrate each network slice and request resource allocation. Moreover, one external *Spirent SPT-N4U network tester* is connected to the server to generate and analyze packets for all traffic flows.[6] In our setup, two physical network interfaces are used to connect such external network tester: one for traffic generation and the other for traffic analysis. In this way, thanks to the specific payload format generated by the network tester, the E2E performance results can be collected.

[5]http://mininet.org/

[6]The tester can be configured to emulate the generation of several flows, each is identified with dedicated MAC, IP addresses, and layer 4 ports.

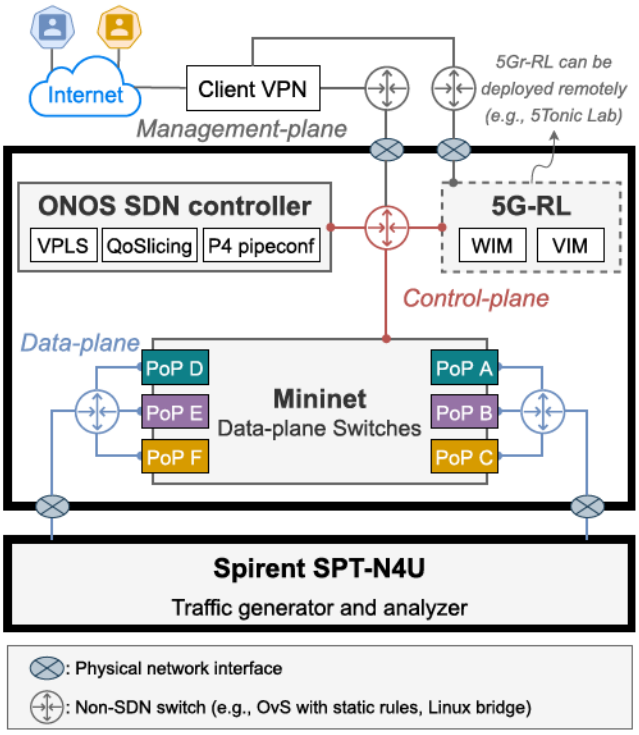


FIGURE 10. Proof-of-concept setup.

Three experiments are then conducted using different resource requests for each slice, as listed in TABLE 2. The first acts as the baseline case without applying any traffic engineering in the data-plane, so there is no performance isolation for any of the network slices. In the second experiment, we apply the proposed performance isolation approach described for the P4 programmable data-plane to drop packets; however, basic packet scheduling is used at the switch's (non-programmable) traffic manager, that is, all packets from each slice are fed into a single physical queue and served according to a First-In-First-Out (FIFO) manner. In contrast, in the last experiment, non-preemptive priority scheduling is applied, with the respective priority levels assigned to each slice as described in TABLE 2, leveraging three physical queues (cf. Fig. 8).

The results of all the experiments are presented in Fig. 11. In *Experiment 1*, the E2E latency is approximately over 800 msec for all slices in Fig. 11a. We note a large queuing delay due to the buffer size, whereas the use of a single FIFO queue for the three slices leads to significant packet drops, as depicted in Fig. 11b. In comparison, after including the virtual queues in the P4 processing pipeline that enable traffic policing - by applying taildrop at the virtual queue and limiting delay either on the physical or virtual queue, the following two experiments only have millisecond-level E2E latency in Fig. 11a. We compare their results in the next paragraph.

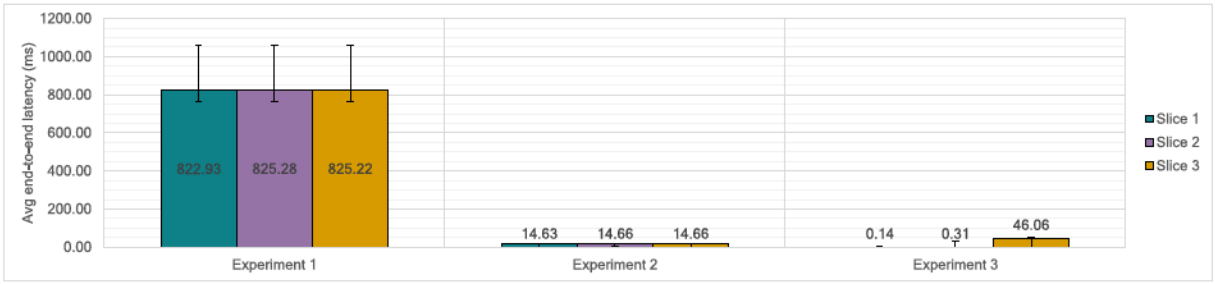
The use of priority queuing can indeed complement the performance isolation. More specifically, before applying priority queuing, the slice with the most stringent delay guarantee (i.e., Slice 1 in TABLE 1) will limit the behaviors of all other slices, just as the similar E2E latency for all three slices shown in *Experiment 2* of Fig. 11a. In comparison, Slice 1 can benefit by reducing its E2E latency and packet drop rate, as depicted in *Experiment 3*. To further dive into the experienced delay within the P4 pipeline, both real delay (in the physical queue) and virtual delay (in the virtual queue) are shown in Fig. 11c. In *Experiment 2*, we can see that all three slices have similar real delays, matching the phenomenon shown in Fig. 11a, determined by the 15 ms delay guarantee for Slice 1. In contrast, priority level determines the type of delay experienced by each slice in *Experiment 3*. Taking Slice 1 and Slice 2 as examples, they only encounter virtual delay. This is because they both fit the bottleneck link capacity. Therefore, in Fig. 11d, their slice-specific AQM drops packets only due to virtual delay, instead of the real delay in Fig. 11c, which is always zero. In comparison, Slice 3 has the lowest priority and cannot be accommodated in the bottleneck link capacity, after the first two prioritized slices. Therefore, the AQM of Slice 3 will drop packets because of the real delay in Fig. 11d, so its delay in P4 switch is guaranteed to be less than 50 msec in Fig. 11c.

TABLE 2. Applied parameters for each slice in three different experiments: (bandwidth guarantee, delay guarantee, priority level).

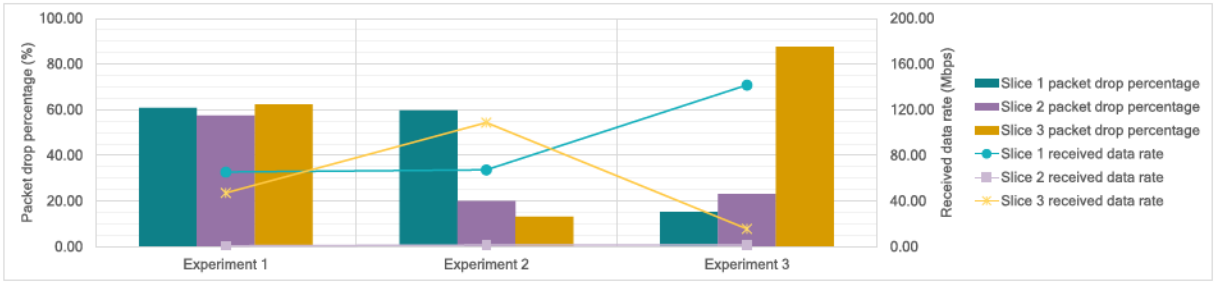
Slice ID	Experiment 1 (No performance isolation)	Experiment 2 (Virtual queue)	Experiment 3 (Virtual queue w/ priority queuing)
1	No slice-specific parameters	138 Mbps, 15 ms, No priority	138 Mbps, 15 ms, High priotiy
2		1.3 Mbps, 30 ms, No priority	1.3 Mbps, 30 ms, Medium priority
3		100 Mbps, 50 ms, No priority	100 Mbps, 50 ms, Low priority

C. DISCUSSIONS

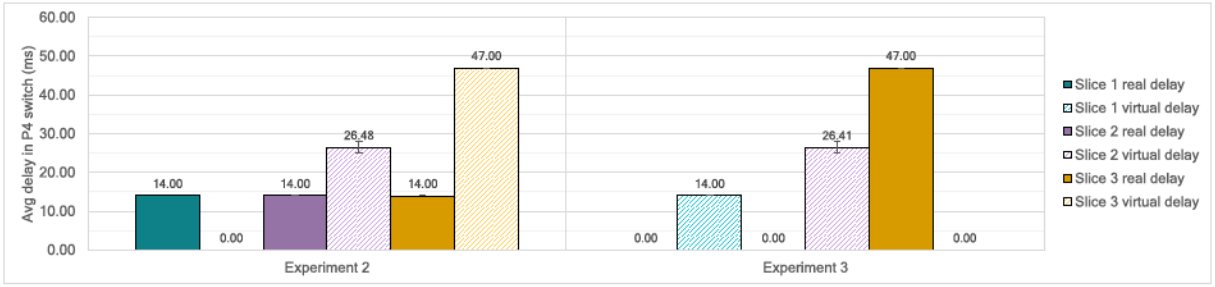
Based on the results shown above, we can notice that the slice-specific AQM in the virtual queue is very important, and its behavior depends largely on the performance guarantee and priority level of each slice. When all slices have the same priority (i.e., *Experiment 2*), they will experience a similar real delay determined by the most stringent delay guarantee, i.e., 15 msec for Slice 1. Hence, the performance of the remaining two slices will be restricted according to their respective bandwidth guarantees (see Table 2) by dropping packets due to the virtual delay in Fig. 11d. On the contrary, Slice 1 will suffer a higher packet drop due to real delay, and we can conclude that the virtual queue is biased against the slice with the most stringent delay guarantee. This syndrome can be remedied if priority levels are assigned in descending order according to the delay guarantee; that is, the lowest priority is assigned to the slice with the most relaxed delay guarantee. Thus, in *Experiment 3*, Slice 3 is limited by its 50 msec delay guarantee in Fig. 11c, whereas Slices 1 and 2 are rate-limited by their respective bandwidth guarantees in Fig. 11b.



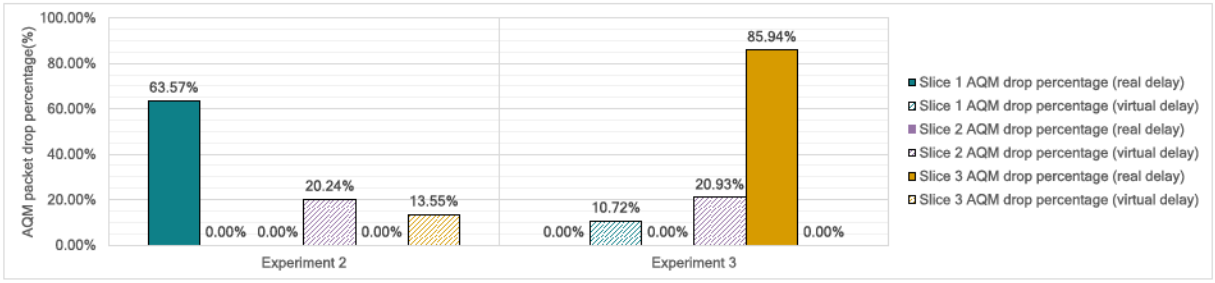
(a) End-to-end one-way latency of each slice



(b) Packet drop percentage and received data rate of each slice



(c) Experienced real delay and virtual delay in P4 switch



(d) AQM packet drop percentage due to real delay or virtual delay in P4 switch

FIGURE 11. Proof-of-concept performance results of different experiments.

Despite the benefits provided by virtual queue over priority queuing, one missing thing is to investigate the performance impact on slices with relaxed delay guarantees because of their lower priority. These slices may have their own traffic adaptation scheme, and therefore the minimum QoE can be maintained (e.g., low-definition video in DASH streaming). However, not all slices have a traffic adaptation scheme, and even if this scheme is adopted, not every packet should be treated equally. For this reason, a future extension is to study programmable traffic management of switches.

Another potential extension would be to study the effects of different bottleneck capacity values. As mentioned in Sec. IV-B, peak capacity cannot always be provisioned after considering wireless channel fluctuations over 5G radio links. Therefore, our solution is expected to be placed next to the 5G CPE (Customer Product Equipment) on the factory floor, connected to the 5G base stations towards the remote office, to provide guaranteed performance for all network slices. By re-examining experiments on smaller bottlenecks, we expect traffic management per flow to provide different levels of reliability for network services in a mobile network. In contrast, an always over-provisioned bottleneck capacity, i.e., much larger than 300 Mbps, will always satisfy the guaranteed throughput with little latency ($<$1 ms).

V. CONCLUSION AND FUTURE DIRECTIONS

In this work, we introduce our solution to ensure performance isolation for network slices in terms of bandwidth and delay guarantees. Specifically, we integrate existing P4 and OpenFlow-based solutions for network slice performance isolation at the transport network data plane, and extend the corresponding 5Growth service architecture and SDN controllers to support, deploy, and configure such solutions. The presented PoC concurrently supports three Industry 4.0 use case scenarios, employing hundreds of flows drawn from the 5Growth Industry 4.0 pilot. The results validate the performance isolation for individual network slices, while investigating the effect of employing different slice traffic priorities, even in a resource-limited situation.

In the future, we plan to extend our work in three directions. The first is to deploy our solution together with private 5G, and recently we have made it functional in 5TONIC laboratory [20] together with a live 5G network in Non-StandAlone (NSA) mode to enable the remote control of Coordinate Measuring Machine (CMM) utilizing a virtual joystick and video cameras. Second, we plan to investigate the impact of specific traffic adaptation schemes on service QoE, especially for delay-sensitive services. In particular, we recently worked on one use case on a live streaming 360-degree video to a mobile device, and future potential use cases are being explored, such as extended reality. Finally, we aim to study the programmability of traffic management as previously tailored in Sec. IV-C.

ACKNOWLEDGMENT

The authors would like to thank Denys Kucherenko and Oleksii Kolodiazhnyi from MIRANTIS for realizing the 5Gr-RL plugins and Koen De Schepper from Nokia Bell Labs for extending the P4 priority queue functionalities, which contributed to the materials presented in this article.

REFERENCES

[1] *Description of Network Slicing Concept, NGMN 5G P1 Requirements Architecture, Work Stream End-to-End Architecture, Version 1.0*, NGMN Alliance, Frankfurt Main, Germany, Jan. 2016.

[2] *System Architecture for the 5G System (5GS): Stage 2 (Release 17)*, document TS 23.501 v17.1.1, 3GPP, Jun. 2021.

[3] *Network Functions Virtualisation (NFV) Release 3: Evolution and Ecosystem: Report on Network Slicing Support With ETSI NFV Architecture Framework (V3.1.1)*, document GR NFV-EVE 012, ETSI, Dec. 2017.

[4] *Management Orchestration: Concepts, Use Cases Requirements (Release 17)*, document TS 28.530 v17.1.0, 3GPP, Mar. 2021.

[5] D. Scano, L. Valcarenghi, K. Kondepu, P. Castoldi, and A. Giorgetti, "Network slicing in SDN networks," in *Proc. 22nd Int. Conf. Transparent Opt. Netw. (ICTON)*, Jul. 2020, pp. 1–4.

[6] *OpenFlow Switch Specification Version 1.5.1*, Open Netw. Found., Menlo Park, CA, USA, Mar. 2015.

[7] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming protocol-independent packet processors," *SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 87–95, Jul. 2014.

[8] C. Papagianni, J. Mangues-Bafalluy, P. Bermudez, S. Barmpounakis, D. De Vleeschauwer, J. Brenes, E. Zeydan, C. Casetti, C. Guimarães, P. Murillo, A. Garcia-Saavedra, D. Corujo, and T. Pepe, "5Growth: AI-driven 5G for automation in vertical industries," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, 2020, pp. 17–22.

[9] The P4 Language Consortium. (May 2017). *P4 16 Language Specification*. Accessed: Nov. 8, 2021. [Online]. Available: https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.html

[10] H. Krishna, N. L. M. van Adrichem, and F. A. Kuipers, "Providing bandwidth guarantees with OpenFlow," in *Proc. Symp. Commun. Veh. Technol. (SCVT)*, Nov. 2016, pp. 1–6.

[11] M. S. Al Breiki, S. Zhou, and Y. R. Luo, "A meter band rate mechanism to improve the native QoS capability of OpenFlow and OpenDaylight," in *Proc. Int. Conf. Adv. Commun. Technol. Netw. (CommNet)*, Apr. 2019, pp. 1–6.

[12] C. Morin, G. Texier, and C.-T. Phan, "On demand QoS with a SDN traffic engineering management (STEM) module," in *Proc. 13th Int. Conf. Netw. Service Manage. (CNSM)*, Nov. 2017, pp. 1–6.

[13] J. Moeyersons, B. Farkiani, B. Bakhshi, S. A. Mirhassani, T. Wauters, B. Volckaert, and F. De Turck, "Enabling emergency flow prioritization in SDN networks," in *Proc. 15th Int. Conf. Netw. Service Manage. (CNSM)*, 2019, pp. 1–8.

[14] J. Heinanen and R. Guerin, *A Two Rate Three Color Marker*, document RFC 2698, 1999.

[15] S. S. W. Lee and K. Chan, "A traffic meter based on a multicolor marker for bandwidth guarantee and priority differentiation in SDN virtual networks," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 3, pp. 1046–1058, Mar. 2019.

[16] Y.-W. Chen, L.-H. Yen, W.-C. Wang, C.-A. Chuang, Y.-S. Liu, and C.-C. Tseng, "P4-enabled bandwidth management," in *Proc. 20th Asia–Pacific Netw. Oper. Manage. Symp. (APNOMS)*, Sep. 2019, pp. 1–5.

[17] N. Zilberman, Y. Audzevich, G. A. Covington, and A. W. Moore, "NetFPGA SUME: Toward 100 Gbps as research commodity," *IEEE Micro*, vol. 34, no. 5, pp. 32–41, Sep./Oct. 2014.

[18] H. Harkous, C. Papagianni, K. De Schepper, M. Jarschel, M. Dimolianis, and R. Preis, "Virtual queues for P4: A poor man's programmable traffic manager," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 3, pp. 2860–2872, Sep. 2021.

[19] C.-Y. Chang, M. A. Jiménez, M. Gharbaoui, J. Sacido, F. Ubaldi, C. Papagianni, A. Zabala, L. Valcarenghi, D. Scano, K. Tomakh, A. Giorgetti, A. Boddi, and K. D. Schepper, "Slice isolation for 5G transport networks," in *Proc. IEEE 7th Int. Conf. Netw. Softw. (NetSoft)*, Dec. 2021, pp. 366–368.

[20] 5TONIC. *5TONIC—An Open Research and Innovation Laboratory dedicated to 5G Technologies*. Accessed: Nov. 8, 2021. [Online]. Available: https://www.5tonic.org/

[21] Fed4Fire. *Fed4FIRE+: Federation for FIRE Plus*. Accessed: Nov. 8, 2021. [Online]. Available: https://www.fed4fire.eu/

[22] *5Growth VS Code Repository*. Accessed: Nov. 8, 2021. [Online]. Available: https://github.com/5growth/5gr-vs

[23] *5Growth SO Code Repository*. Accessed: Nov. 8, 2021. [Online]. Available: https://github.com/5growth/5gr-so

[24] *5Growth RL Code Repository*. Accessed: Nov. 8, 2021. [Online]. Available: https://github.com/5growth/5gr-rl

[25] *5Growth VoMS Code Repository*. Accessed: Nov. 8, 2021. [Online]. Available: https://github.com/5growth/5gr-mon

[26] *5Growth AIMLP Code Repository*. Accessed: Nov. 8, 2021. [Online]. Available: https://github.com/5growth/aimlp

[27] *Network Functions Virtualisation (NFV): Management and Orchestration: Or-Vi reference Point–Interface and Information Model Specification (V2.1.1)*, document GS NFV-IFA 005, ETSI, Apr. 2016.

[28] *Container Network Interface—Networking for Linux Containers*. Accessed: Nov. 8, 2021. [Online]. Available: https://github.com/containernetworking/cni

[29] *Multus CNI Code Repository*. Accessed: Nov. 8, 2021. [Online]. Available: https://github.com/intel/multus-cni

[30] *Service Requirements for Cyber-Physical Control Application Vertical Domains: Stage 1 (Release 16)*, document TS 22.104, 3GPP, Dec. 2020.

[31] *A 5G Traffic Model for Industrial Use Cases*, 5G-ACIA, Frankfurt Main, Germany Dec. 2019.

[32] X. Li, A. Garcia-Saavedra, X. Costa-Perez, C. J. Bernardos, C. Guimaraes, K. Antevski, J. Mangues-Bafalluy, J. Baranda, E. Zeydan, D. Corujo, P. Iovanna, G. Landi, J. Alonso, P. Paixao, H. Martins, M. Lorenzo, J. Ordonez-Lucena, and D. R. Lopez, "5Growth: An end-to-end service platform for automated deployment and management of vertical services over 5G networks," *IEEE Commun. Mag.*, vol. 59, no. 3, pp. 84–90, Mar. 2021.

[33] I. Markopoulos, L. Valcarenghi, I. Mesogiti, M. Taferner, P. Boleguin, F. Bouali, K. Tzanettis, I. Tzanettis, D. Hetzer, and I. Känsälä, "Service performance measurement methods over 5G experimental networks," 5GPPP, EURESCOM, Heidelberg, Germany, White Paper ICT-19 Performance KPIs, May 2021. [Online]. Available: https://zenodo.org/record/4748385, doi: 10.5281/zenodo.4748385.
[34] C. Guimaraes, X. Li, C. Papagianni, J. Mangues-Bafalluy, L. M. Contreras, A. Garcia-Saavedra, J. Brenes, D. S. Cristobal, J. Alonso, A. Zabala, J.-P. Kainulainen, A. Mourad, M. Lorenzo, and C. J. Bernardos, "Public and non-public network integration for 5Growth industry 4.0 use cases," *IEEE Commun. Mag.*, vol. 59, no. 7, pp. 108–114, Jul. 2021.
[35] C.-Y. Chang and N. Nikaein, "Closing in on 5G control apps: Enabling multiservice programmability in a disaggregated radio access network," *IEEE Veh. Technol. Mag.*, vol. 13, no. 4, pp. 80–93, Dec. 2018.

CHIA-YU CHANG (Member, IEEE) received the Ph.D. degree from Sorbonne Université, France, in 2018. He is currently a Network System Researcher at Nokia Bell Labs, Belgium. He has had over ten years of experience in algorithm/protocol research on mobile communication systems since the 3G era. He has extensive research experience at both academic and industrial laboratories, including EURECOM Research Institute, MediaTek, Huawei Swedish Research Center, and Nokia Bell Labs. He has participated in several EU collaborative research and innovation projects in the ICT field, such as COHERENT, Q4Health, SLICENET, 5G-PICTURE, 5Growth, and DAEMON. His research interests include wireless communication, computer networking, and edge computing, with particular interest in 5G and beyond, network slicing, and traffic engineering.

TERESA GINER RUIZ received the B.S. degree in telecommunication technologies engineering from the Universidad de Zaragoza, in 2019, and the M.S. degree in telematic services and network engineering from the Universidad Politécnica de Madrid, in 2021. She is currently a Network Engineer and a Researcher at Telcaria Ideas SL, Spain. She has been involved in the EU H2020 Project 5Growth. Her main research interests include software defined networks (SDN), network function virtualization (NFV), and 5G networks.

FRANCESCO PAOLUCCI is currently a Senior Researcher at CNIT, Pisa, Italy. He has been involved in many industrial and European research projects on next generation networking control. He is the coauthor of three IETF Internet Drafts, more than 180 publications in international journals, conference proceedings, book chapters, and filed four international patents. His main research interests include the field of network control plane and service orchestration for edge platforms, traffic engineering, network disaggregation, advanced monitoring/telemetry, and SDN data plane programmability. He is an Associate Editor of the IEEE/OSA Journal of Optical Communications and Networking (JOCN) and an Executive Editor of the *Transactions on Emerging Telecommunications Technologies* (ETT).

MANUEL A. JIMÉNEZ received the degree in telecommunications technologies engineering from the Universidad de Sevilla, in 2020. He is currently a Software Engineer at Capgemini Engineering and formerly a Network Engineer and a Researcher at Telcaria Ideas SL, Spain. He has been involved in 5Growth and at Cloud Core Policy Controller for Ericsson. His development interests include 5G and network slicing.

JAVIER SACIDO received the B.S. degree in computer science and engineering and the M.S. degree in informatics engineering from the Universidad Carlos III de Madrid, in 2018 and 2020, respectively. He is currently a Network Engineer and a Researcher at Telcaria Ideas SL, Spain. He has been involved in different EU H2020 projects, like 5G-Dive and 5Growth. His main research interests include artificial intelligence, software-defined networks, and 5G networking.

CHRYSA PAPAGIANNI (Member, IEEE) is currently an Assistant Professor at the Informatics Institute, University of Amsterdam. She is a part of the Multiscale Networked Systems Group that focuses its research on network programmability and data-centric automation. Prior to joining UvA, she was a Network Research Engineer at Bell Labs, Antwerp, as part of the end-to-end Network Service Automation Laboratory. From 2016 to 2018, she was a Research Scientist at the Institute for Systems Research, University of Maryland, USA. Her research interests include programmable networks with emphasis on network optimization and the use of machine learning in networking. She has participated in various EU FIRE and 5G-PPP projects, such as Fed4FIRE+, OpenLab, NOVI, and 5Growth, working on issues related to network slicing.

FABIO UBALDI received the Dr.Ing. degree in telecommunications engineering from the University of Perugia, Italy, in 2001. He was in a consortium research on telecommunication (CoRiTeL), from 2002 to 2011. He is currently a Senior Research in control plane methods for optical and radio system at Ericsson Research, Italy. In about 20 years of professional experience, his research activity encompassed several aspects of controls plane of packet, optical, and radio telecommunication systems, including software defined networking (SDN) and management and network orchestration (MANO). He has contributed to research projects in all these areas, in collaboration with worldwide recognized universities and research institutions. He has coauthored 16 filed patent applications and more than ten publications on optical networks.

DAVIDE SCANO received the B.S. degree in telecommunication engineering from the University of Pisa, in 2017, and the M.S. degree in computer science and networking from Pisa University and Scuola Superiore Sant'Anna, Pisa, in 2019, with a research thesis on SDN for guarantee QoS in network slicing. He is currently pursuing the Ph.D. degree in emerging digital technologies with Scuola Superiore Sant'Anna. His research interests include software defined networking, next generation software defined networking, optical networks, and disaggregated networks. In 2020, he got a Research Scholarship at Scuola Superiore Sant'Anna.

MOLKA GHARBAOUI received the Ph.D. degree in innovative technologies of information and communications engineering and robotics from Scuola Superiore Sant'anna, Pisa, in 2012. She is currently a Researcher Affiliate at CNIT, Italy. She has been involved in several EU projects and has coauthored more than 50 papers appeared in international journals and conferences. Her main research interests include software-defined networking, network function virtualization, network orchestration, and the development and the implementation of service-oriented architectures and service management for smart cities.

ALESSIO GIORGETTI received the Ph.D. degree from Scuola Superiore Sant'Anna (SSSA), Pisa, Italy, in 2006. In 2007, he was a Visiting Scholar at the Centre for Advanced Photonics and Electronics, University of Cambridge, U.K. He was an Assistant Professor at SSSA, from 2007 to 2020. He is currently a Researcher at IEIIT-CNR, Italy. He is an Active Software Contributor to Open Network Foundation projects. He is the author of more than 100 publications, including international journal articles, conference proceedings, and patents. His research interests include optical network architectures and control plane, industrial networks design, software-defined networking, and quantum communications.

LUCA VALCARENGHI (Senior Member, IEEE) has been an Associate Professor at Scuola Superiore Sant'Anna (SSSA), Pisa, Italy, since 2014. He has published almost 300 papers (source Google Scholar, in May 2020) in international journals and conference proceedings. His main research interests include optical networks design, analysis, and optimization; communication networks reliability; energy efficiency in communications networks; optical access networks; zero touch network and service management; experiential networked intelligence; and 5G technologies and beyond. He received a Fulbright Research Scholar Fellowship, in 2009, and a JSPS "Invitation Fellowship Program for Research in Japan (Long Term)," in 2013.

KONSTANTIN TOMAKH is currently a Network Software Engineer at Mirantis, Ukraine. He has more than 15 years of experience in the IT, network administration, and telecom industries. He is focused on OpenStack networking, container networking, SDN, NFV and edge cloud and private cloud building solutions, and Python and Java software development. He holds number of industry certifications, including Cisco CCNP, Kubernetes, and AWS.

ANDREA BODDI received the master's degree in computer engineering from the University of Florence, Italy, in 2020. He is currently a Researcher in control plane methods for optical and radio system at Ericsson TEI. His research interests include 5G, software defined networking (SDN), management and network orchestration (MANO), and machine learning applied to optical and radio telecommunication networks.

AGUSTÍN CAPARRÓS received the Ph.D. degree from Madrid Technical University, in 2018. He has worked in the internetworking industry as a Product Support Engineer, a Network Deployment Engineer, and a Consultant Engineer. He is currently a Network Engineer and a Senior Research Engineer at Telcaria Ideas SL, Spain. He is involved in 5Growth EU Project. His research interests include diverse, focusing on SDN/NFV, edge platforms, and network monitoring and telemetry.

MATTEO PERGOLESI received the master's degree in computer and automation engineering and the Ph.D. degree in industrial and information engineering from the University of Perugia, in 2015 and 2019, respectively. He is currently a Chief Research Officer and a VP of engineering at Telcaria Ideas S.L. His research interests include 5G, network function virtualization, and edge computing.

BARBARA MARTINI is currently a Research and Development Manager and the Head of research at CNIT, Italy. Before joining CNIT PNT Laboratory, she worked for two large telco companies, Italtel and Marconi Communications (currently Ericsson). She is an Adjunct Professor with the Sant'Anna School of Advanced Studies and the University of Pisa, Italy. She has been involved in several national/EU research projects, the recent ones 5GPPP 5GEX, 5GTRANSFORMER, and 5GROWTH, and in several FIRE projects (OFELIA, FED4FIRE+, TRIANGLE, and 5GINFIRE) with leading roles. She has coauthored more than 100 papers in international journals and conference proceedings. Her research interests include network virtualization and orchestration in SDN/NFV/5G environments, service platforms for next-generation networks, network control/management architectures, and security solutions for multi-domain IP/optical net-works and NFV deployments.