IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 29, 2020, accepted January 4, 2021, date of publication January 18, 2021, date of current version January 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3051940

Adaptive Network Slicing in Multi-Tenant 5G IoT Networks

ANTONIO MATENCIO ESCOLAR[1], JOSE M. ALCARAZ-CALERO[1], (Senior Member, IEEE), PABLO SALVA-GARCIA[1], JORGE BERNAL BERNABE[2], AND QI WANG[3]

[1]School of Computing, Engineering and Physical Sciences, University of the West of Scotland, Paisley PA1 1LU, U.K.
[2]Department of Information and Communication Engineering, University of Murcia, 30100 Murcia, Spain
[3]School of Computing, Engineering and Physical Sciences, University of the West of Scotland, Paisley PA1 1LU, U.K.

Corresponding author: Jose M. Alcaraz-Calero (jose.alcaraz-calero@uws.ac.uk)

This work was supported in part by the European Commission through the project 6G Bringing Reinforcement learning Into Radio Light Network for Massive Connections (BRAINS) under Grant H2020-ICT-2020-2/101017226, and in part by the University of the West of Scotland 5G (UWS 5G) Video Laboratory project.

ABSTRACT The Fifth Generation (5G) mobile networking coupled with Internet of Things (IoT) can provide innovative solutions for a wide range of uses cases. The flexibility of virtualized, softwarized and multi-tenant infrastructures and the high performance promised by 5G technology are key to cope with the deployment of the IoT use cases demanded by various vertical businesses. Such 5G IoT use cases incur challenging Quality of Service (QoS) requirements especially connectivity for millions of IoT devices to achieve massive Machine-Type Communication (mMTC). In addition, network slicing is a key enabling technology in 5G multi-tenant networks to create logical virtualized networks for delivering customised solutions to meet diverse QoS requirements. This work presents a 5G IoT framework with network slicing capabilities able to manage a vast number of heterogeneous IoT network slices dynamically on demand. The proposed solution has been empirically tested and validated in five realistic vertical-oriented IoT use cases. The achieved results demonstrate a excellent stability, isolation and scalability while being able to meet extreme QoS requirements even in the most congested and stressful scenarios.

INDEX TERMS 5G, Internet of Things (IoT), network management, network slicing, quality of service (QoS), software defined networks (SDN).

I. INTRODUCTION

The Fifth Generation (5G) mobile networks allow instantiating multiple concurrent logical systems to meet diverse vertical industry service requirements, over a common underlying softwarized, virtualized and multi-tenant infrastructure. The Internet of Things (IoT) is being embraced by the 5G architecture, where various vertical IoT services such as Industrial IoT (IIoT), Smart Agriculture or Smart Cities among others, can be instantiated over the same shared physical network and cloudified 5G infrastructure.

Each vertical imposes different requirements in terms of network Quality of Service (QoS) parameters such as bandwidth, reliability and latency/delay. For instance, IIoT typically requires high reliability to cope with critical services based on Ultra-Reliable and Low-Latency Communications (uRLLC). Smart agriculture scenarios may need Massive Machine-Type Communication (mMTC), and demand efficient communications with millions of constrained IoT devices. Some use cases in Smart cities e.g., traffic monitoring through video cameras would need Enhanced Mobile Broadband (eMBB) communication to achieve high data rates and low latency.

To meet the diverging network performance demands by heterogeneous vertical use cases, network slicing has emerged as a key approach in 5G. A network slice can be defined as a set of network functions and the associated network resources that are logically isolated and allocated according to the Service-Level Agreement (SLA), and can be shared per kind of traffic, per service, protocol or application, or dedicated per user or per device, if allowed by a flexible 5G management system. Different types of communications can be supported by different kinds of network slices [1].

Therefore, different IoT services can be virtualized and instantiated in dedicated slices based on different service requirements, where IoT service providers can allocate those

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

slices in a cost-effective way and attend to users preferences. Industrial IoT services, e.g. mission critical services, can be moved to the Edge of the network in a specific slice to minimize transition delay while massive IoT services e.g. for smart cities will require vast number of dedicated slices, each one with different QoS needs. As recently highlighted as major challenges in [2], 5G network slicing should enable new business models for delivering heterogeneous 5G services exploited by different industry verticals, combining different 5G services types including mMTC, uRLLC and eMBB while ensuring necessary performance and scalability through proper slice isolation and dynamism.

To realize a truly scalable 5G IoT slice management, there could be multiple concurrent shared slices plus a potential vast number of dedicated slices allocated per IoT device, with different network parameters needs, and with different packet sizes and protocols. In the worst case, this could mean that the virtual switches in the datapath need to cope with millions of heterogeneous concurrent adaptive slices.

5G management frameworks and network elements, such as virtual network switches and routers deployed over the 5G infrastructure need to be evolved to control dynamically the network traffic of multiple, heterogeneous and concurrent tenants, from diverse verticals, ensuring QoS parameters over the shared network's available resources. The new imposed requirements are manifold, as the new 5G slicing implementations need to deal with nested-encapsulation in the datapath to support both mobility and multi-tenancy typically through tunnelling protocols such as VXLAN and GTP, support scalability to handle a vast number of concurrent heterogeneous slices, enable flexibility and elasticity to manage and orchestrate dynamically on demand slicing rules over multiple virtual network appliances and thus ensure the slices end-to-end, and assure interoperability with enriched slicing models enforceable and understandable in the data plane.

To the best of the author's knowledge, the solutions identified in the current literature provide either partial solutions for specific use cases or present preliminary outcomes. They have not been validated in a realistic 5G IoT scenario where different services with varying QoS requirements and traffic profile are demanded, challenging the 5G infrastructure and imposing the imperative of addressing 5G network traffic with multiple levels of nested encapsulation.

To contribute to addressing the above challenges, this paper introduces a new 5G IoT slice control framework that allows the adaptive instantiation and management of heterogeneous and evolving slices over multiple technological IoT domains, which demand different slicing according to the vertical needs.

The contributions of this paper are manifold:

- A practical cognitive network management framework is presented for autonomic enforcement of scalable network slicing in 5G IoT networks.
- A new, highly scalable network slicing approach is proposed for multi-tenant 5G networks especially devised for virtualized, multi-tenant, and mMTC communications, managing thousands of heterogeneous slices.
- A new enforceable protocol is defined to apply network slices in the 5G IoT network and cope with in a homogeneous and technology-agnostic approach with the different technologies and hardware composing the data plane.
- The network slicing approach has been implemented in the data plane by significantly extending Open vSwitch (OVS), using a kernel space mechanism in order to have full control of the slicing in virtualized 5G IoT networks.
- Agile and flexible management of slices which allows dynamic adaptation on demand to the verticals' needs.
- The proposed solution has been empirically validated, with performance evaluation over a real 5G IoT network deployment using vertical-oriented use cases.

The rest of the paper is organized as follows. In Section II we analyze the current state of the art regarding network slicing in 5G IoT networks. Section III lays out the specific requisites that 5G IoT networks must satisfy to provide adaptive network slicing capabilities and provides an overview of the proposed 5G IoT multi-tenant architecture. Section IV presents the proposed Slice Control architecture. Then, Section V introduces the vertical-oriented IoT use cases used to empirically evaluate and validate the solution proposed in this work. Section VI provides implementation details and a description of the testbed deployed to conduct the experiments. Section VII presents the achieved experimental results in terms of scalability, QoS performance and adaptive management of the life cycle of network slices. Finally, conclusions are drawn with future research activities outlined in Section VIII.

II. BACKGROUND AND RELATED WORK

Network slicing for 5G IoT deployments plays a key role to provide the capability to enable the new business models expected in the IoT era. This fact is evidenced by a significant amount of research activity and contributions on this subject.

In a recent survey, Khan *et al.* [3] review the latest advances of network slicing for IoT verticals and use cases, including smart transport systems, smart industry, smart homes and smart care. They identify scalability, interoperability and efficient resource allocation among the major open research challenges relevant to network slicing. Furthermore, the authors highlight as one of their main conclusions that network slicing is an indispensable technology to enable a wide range of 5G use cases and beyond. They also underline that it is necessary to propose novel adaptive network slicing management schemes to cope on demand with services of varying users' demands.

Several authors have proposed 5G-based solutions for specific IoT use cases. For instance, Cheng *et al.* [4] propose a 5G IoT architecture for smart manufacturing although no implementation is provided. In [5], the authors analyze how network slicing technology in 5G networks can meet the specific needs of different smart grid services to support multiple power business scenarios in Power Internet of Things (PIoT). In [6], the authors focus on IoT solutions for Smart Home applications and propose a 5G architecture where three slices are deployed (Smart Home security, eMBB traffic and Massive IoT traffic). Unlike our proposal that provides a 5G IoT framework able to cope with heterogeneous 5G IoT traffic, all these work address the deployment of IoT use cases in 5G infrastructures from the perspective of a given single use case and the authors do not provide an empirical validation and evaluation of the offered solution.

Kapassa *et al.* [7] describe an IoT-oriented architecture for 5G network slicing that enables the allocation of Quality of Service (QoS) of diverse concurrent IoT applications and services with different requirements. Nonetheless, they do not implement or validate their proposal.

In [8], the authors propose an authentication framework supporting network slicing and fog computing for 5G IoT, allowing users to establish secure connections with IoT devices through dedicated slices. Unlike their work, we focus on the mechanism for slice management in the 5G datapath, rather than security aspects of slicing.

Kurtz *et al.* [9] propose and implement a slicing mechanism for 5G networks based on SDN/NFV, evaluated in demanding critical infrastructure use cases. They use OVS and SDN controllers to enforce the slices. However, they do not use real 5G traffic that demands specific modifications (e.g. due to encapsulation to provide tenant isolation and user mobility in 5G networks) in the datapath management (e.g. adaptation in switches like OVS) to handle the slices.

In [10], the authors propose an end-to-end mIoT slicing mechanism with enhanced control and user planes (CP/UP) for the 5G network. Unlike in our work that focuses on ensuring the QoS in the datapath, they focus on reducing the signaling overhead and an efficient UP resource utilization. Moreover, their solution is still preliminary as it is simulated not implemented in a realistic 5G network.

Bektas *et al.* [11] propose a RAN network slicing scheduler for 5G, aimed to deal with mission critical services demanded by IoT verticals (e.g. Smart Grid).

In [12], the authors propose and validate dynamic network slicing for 5G IoT and eMBB services. Although they manage properly and dynamically the slices, their proposal assumes just two slices for the traffic profile, i.e. one for IoT traffic and the other for eMBB services. Similarly, the slicing architecture proposed by [13] evaluates one slice per 5G services types: mMTC, eMBB, or uRLLC.

A software-based data plane programmability approach is followed by Salva-Garcia *et al.* [14] to perform network traffic filtering for multi-tenant 5G IoT networks in kernel space. Like in our work, they follow an intent-based and SDN model for adaptive, flexible and network filtering in massive MTC scenarios demanded by IoT. Similarly, Matencio-Escolar *et al.* [15] implement a software-defined firewall for 5G NB-IoT networks to increase considerably the number of supported rules (up to one million), which makes it fully suitable for highly demanded mMTC services demanded by IoT scenarios. Despite these two work address the mMTC services requirements, they are not focused on QoS management needed for dealing with 5G network slicing.

A novel network slicing framework for 5G networks is presented and evaluated in [16]. The authors address specific challenges of media use cases, through an End-to-End QoS-aware slicing intended to support migration of mission critical services (eHealth tele-medicine services) to 5G networks. The proposed system implements Low Latency MEC Platform, with QoS control based on data plane programmability that allows establishing priorities of the 5G network traffic. Unlike our work which supports combination of either mMTC, eMBB, and URLLC services, they focus mainly on media use cases, with high demands in terms of low latency (URLL services) and high bandwidth (eMBB).

Regarding network slicing solutions complying with the specifications and standards developed by the Standardization Development Organizations (SDOs) and the 5G Industry, Diaz-Rivera *et al.* [17] propose a Network Slice Selection Function (NSSF) to enable the selection of slices in the data plane meeting the 3GPP functionality specifications [18]. The authors provide a functional validation although they do not report results on scalability or behaviour in IoT environments or 5G multi-tenant networks. This paper highlights the importance of providing an abstraction mechanism for network configuration due to the growing complexity of the data plane which is one of the strengths of our contribution (see *Data Plane Abstraction Service* description in section IV).

Network Slicing in industrial environments with extreme QoS requirements is a major challenge faced by the 5G industry and several contributions have been made in the recent past. In this regard, in [19] authors propose a method to perform network slicing for deterministic and packet-switched industrial communication protocols such as *OPC-UA*. Rost *et al.* [20] describe a testbed in a real industrial environment, the Hamburg port area, where they deploy a 5G infrastructure with network slicing capabilities such as slice isolation, flexible slice customization and multi-tenancy. In [21], authors propose a framework aimed at providing network slicing capabilities to meet the specific requirements of Industry 4.0. The framework provides slicing through public and private federated infrastructures across Radio, Edge and Core segments. The framework is validated with 3 use cases: remote production monitoring, remote equipment maintenance and dynamic industrial manufacturing. Although promising, none of these works give results on scalability and performance and therefore do not demonstrate their suitability in the foreseen large-scale industrial IoT scenarios. They also do not cover multiple use case running simultaneously to demonstrate the suitability of slicing.

Efficient QoS-aware data plane network slicing in 5G networks has been recently implemented in hardware by Ricart-Sanchez *et al.* [22]. It provides hardware-based traffic classification, priority configuration, and traffic scheduling using NetFPGAs. Moreover, the framework deals with traffic with two levels of encapsulation (VXLAN, GTP) to support multi-tenant 5G networks and allows performing isolation for 5G MEC network traffic. Although the general goal of that paper is similar to ours, they do not address the particularities of IoT scenarios, with thousands of devices and slices, as evaluated in our proposal.

Despite the considerable number of related work in the area of 5G IoT network slicing, practical, scalable and flexible software-defined networking solutions are still not sufficiently researched from the 5G data path perspective to satisfy various use cases of diverging requirements including not only massive IoT scenarios but also hybrid scenarios with heterogeneous IoT traffic profiles.

III. NETWORK SLICING IN VIRTUALIZED AND MULTI-TENANT 5G IOT DEPLOYMENTS

A. SLICING REQUIREMENTS IN 5G-ENABLED IoT NETWORKS

There are a number of specific requirements for adaptive network slicing in 5G IoT networks, listed as follows:

- **Multi-tenant support**: 5G IoT networks comprise virtualized network functions running over a shared physical infrastructure, which is exploited simultaneously by different verticals and operators. Therefore, the network slicing system should be able to handle encapsulated network traffic intended to differentiate the network slices and the associated SLAs per each tenant.
- **Mobility support**: as identified as a major challenge in 5G IoT slicing [2], the slicing system should be able to handle, differentiate, manage and control efficiently the GTP encapsulated network traffic needed to deal with User Equipment (UE) i.e., IoT device mobility. This includes handling the Uplink/Downlink traffic differentiating tunnels by the management framework and the slicing agent. Indeed, the slicing system should support nested encapsulation to deal with mobility and multi-tenancy simultaneously.
- **Scalable slicing for mMTC**: in order to support mMTC services, the slicing system should be able to handle and differentiate traffic coming from a vast number (millions) of different devices. It could imply managing and control efficiently thousands of concurrent network slices (in the most demanding case one slice per IoT device).
- **IoT service differentiation with high bandwidth requirements (eMBB services)**: certain IoT verticals (e.g. video surveillance in Smart cities) might require broadband communications with high transmissions rates. The slicing management system should support subscription associated to high bandwidth requirements, and the slicing control system should be efficiently handle that kind of traffic.
- **IoT device differentiation with reliable and low latency requirements (URLLC services)**: the slicing system should support the subscription, management and control of dedicated slices with ultra reliable low latency requirements.
- **Heterogeneous slicing support:** the slicing system should concurrently and efficiently support the control and management of heterogeneous network traffic and protocols demanded by different verticals, who may use any of the 5G services types including mMTC, URLLC and eMBB or a combination of these types in more demanding use cases.
- **Flexible and adaptive management**: the slicing system should be able to manage on demand and efficiently adapting the slicing of the evolving 5G IoT network traffic, and adding or decommissioning the slicing policies. This will allow upper cognitive layers in the 5G management framework to automatically update the network slicing behaviour according to the context.
- **Interoperability**: the system should be managed by a common, interoperable and high-level abstract interface, following a common data model intended to handling dynamically different kinds of slices for heterogeneous traffic.

B. 5G IoT MULTI-TENANT INFRASTRUCTURE FOR VERTICAL BUSINESSES

This section overviews an infrastructure that perfectly matches the case of study in this paper and gives a realistic scenario to research and evaluate the proposed solution. From a bottom-up perspective, Fig. 1 shows four different network segments: *(i)* The Radio Access Network to connect individual devices to the 5G network. *(ii)* The Mobile/Multi-access Edge Computing (MEC) segment to enhance the experience of end users when accessing common services through computing resource localization. *(iii)* The 5G Core Network segment with expanded service capabilities, scalability, agility and network functions. *(iv)* The Inter-domain Segment to reach the Internet and other domains managed by different organisations.

It is noted that the 5G architecture follows a Service Based Architecture (SBA) and builds upon virtualization and softwarisation paradigms. These technologies are expected to have a significant impact on 5G deployment as they aim to offer significant agility and flexibility to meet vertical industries' service requirements while providing multi-tenancy and user mobility capabilities in a common underlying infrastructure. That allows scenarios like the one presented in Fig. 1 where different verticals, each one imposing different network requirements (e.g., eMBB, mMTC, uRLLC), are co-existing in the same infrastructure. To deal with such heterogeneity of the network, this paper proposes a Software-Defined Network (SDN) approach that is based on extending the industry-leading, production-quality OVS platform and allows controlling, managing and acting on complex network traffic (see Fig. 4) traversing these architectures. The objective is to guarantee the effective functioning of the vertical slices associated to each tenant/service through the whole 5G network. Specifically, we leverage the proposed 5G IoT Slice Management Framework to interact with this enhanced Open vSwitch to ensure that SLAs are respected.

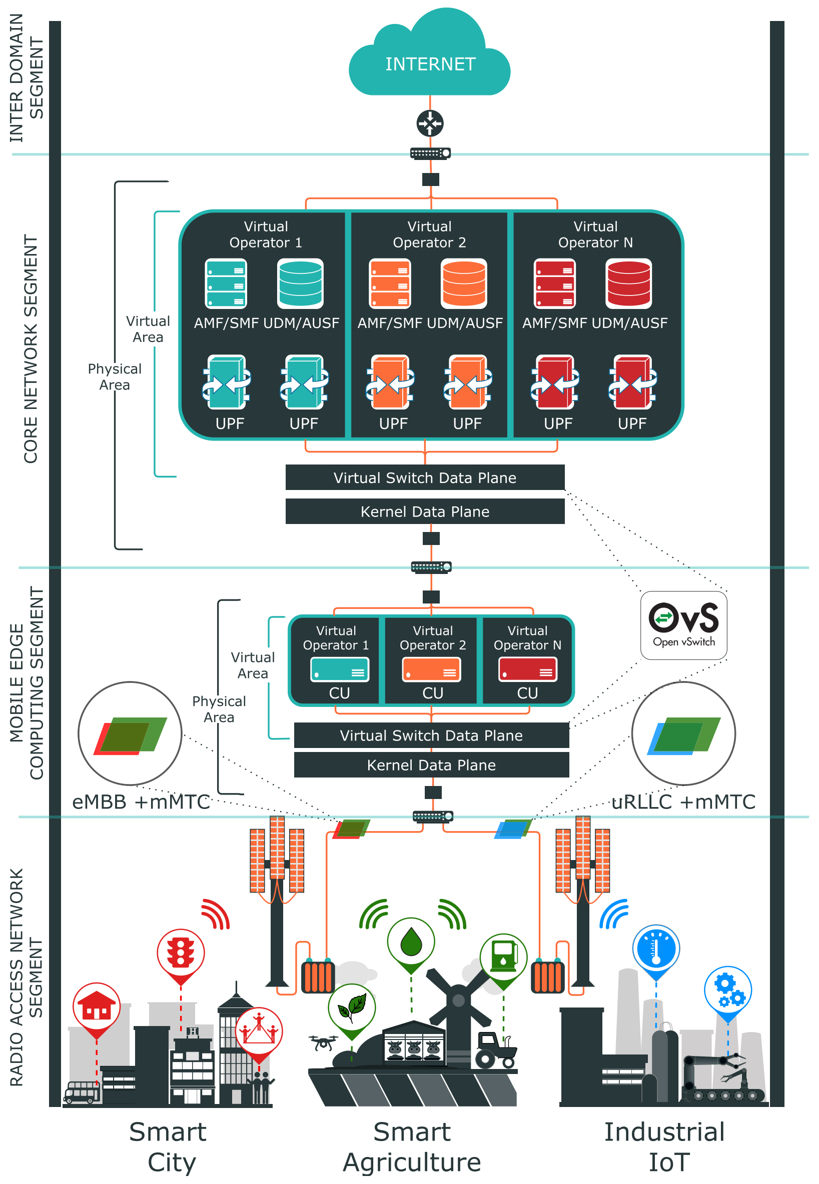


FIGURE 1. Overview of the proposed 5G IoT Multi-Tenant Infrastructure.

C. 5G IoT SLICE MANAGEMENT FRAMEWORK

To be able to provide dynamic adaptation of the network slicing capabilities presented in this research, a Slice Management Framework has been adopted as illustrated in Fig. 2. The framework is composed mainly of five architectural domains. The arrows in Fig. 2 indicate the communication management among the different planes and their underlying modules. Roles and responsibilities of each component are described below, from a bottom-up perspective.

1) INFRASTRUCTURE DOMAIN

- **Physical Area**. It contains all the 5G hardware resources such as computes, storage and networking devices, among others, placed throughout the physical infrastructure able to be shared among the different tenants.
- **Virtual Area**. It contains all the virtualised equipment (created by the hypervisor) running on top of the physical computers that represent logical hardware resources assigned to specific tenants.

2) CONTROL AND MANAGEMENT DOMAIN

- **Sensing**. It provides a common point to manage all the different sensors that report metrics about the current state of the infrastructure. Here, time-based metric reports are collected for monitoring purposes such as port speed, bandwidth consumed or bandwidth available at different points of the network.
- **SDN Controller**. It is the module that maintains a strategic control point in an SDN as the one presented here. From its northbound API (NBI) it receives the

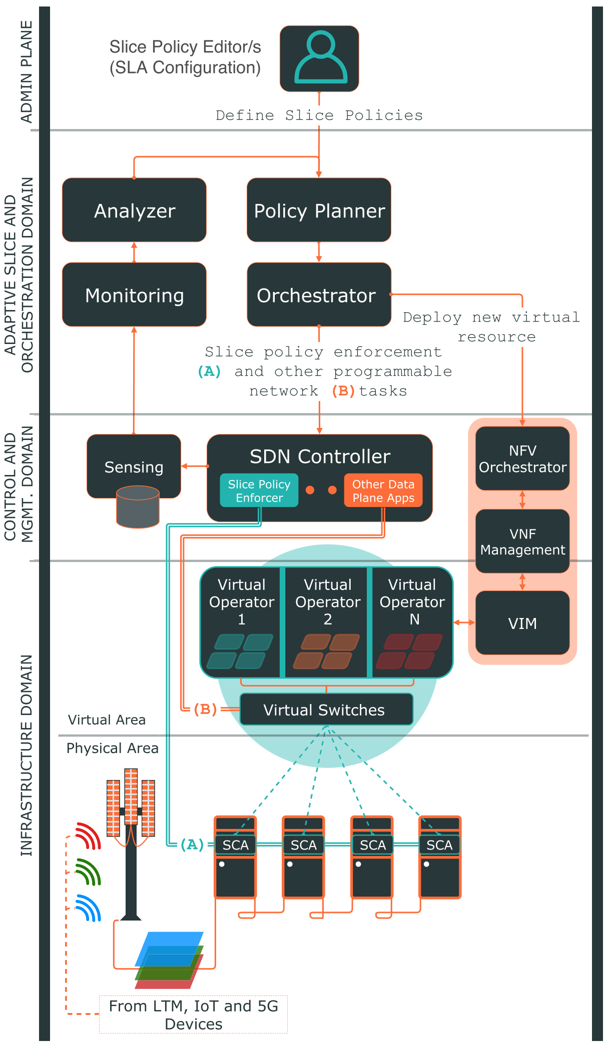


FIGURE 2. Architecture of the cognitive 5G IoT Slice Management Framework.

instructions to apply the business logic by controlling flows and programming networking devices placed on the data plane. In particular, this research proposes a new component in the data plane, the Slice Control Agent (SCA), which will finally be in charge of such programming of the network device. This is further explained with more detail in Section IV-A.

- **VNF Orchestrator**. It is in charge of the deployment of the different Virtual Network Functions (VNFs) by considering current physical and logical resources of the infrastructure.
- **VNF Manager**. It is a key component of the Management and Orchestration paradigm that works in concert with the VNF Orchestrator (VNFO) and/or the Virtual Infrastructure Manager (VIM) and it is in charge of controlling, managing and monitoring the VNFs' life cycle. In addition, it also controls Element Management System (EMS) and/or Network Management System (NMS).
- **Virtual Infrastructure Manager (VIM)**. It controls virtual resources and the life cycle of all the virtual hardware resources in the virtual infrastructure. It also keeps the catalog and the inventory of the existing VMs.

3) ADAPTIVE SLICE AND ORCHESTRATION DOMAIN

- **Monitoring**. It monitors metrics provided by the Sensing module to perform highly scalable data storage.
- **Analyzer**. It allows the definition of rules that are used to monitor the current status of the infrastructure by using spatial and temporal correlation in the information gathered from both the monitoring component and the resource inventory. The rules produce alerts about specific events in the infrastructure that require attention.
- **Policy Planner**. It makes it possible to define templates that are used to transform the strategy indicated by the Slice Policy Editor into an ordered set of implementable steps required to implement the strategy received.
- **Orchestrator**. It receives an implementable plan and then orchestrates the execution of each of the steps involved in the plan in order to enforce it into the infrastructure with the purpose of resolving the alert.

4) ADMIN PLANE

- **Slice Policy Editor**. Any authorized external agent that can reach the exposed functions by the framework to create, modify or remove network slices. Usually, vertical services administrators will define here their tailored configurations.

IV. SLICE CONTROL ARCHITECTURE

Fig. 3 depicts a modular architecture divided in three different planes. First (on top), the **Infrastructure Management Plane** provides the slice policies to be enforced throughout the infrastructure. Next, the **Infrastructure Control Plane** is used as an entry point to apply traffic control rules to any of the programmable networking devices available on the network equipment by following the SDN principles. It is noted that usually this plane directly programs requested network control actions into a concrete networking device. However, as it is shown in this figure, we delegate this task to a proposed Slice Control Agent (SCA) (at the Resource Control Plane), to abstract particular slicing tasks and to homogenize technology-dependant syntax among different Flow Agents that co-exist in the network. To this end, at the lower end of the architecture, it is placed in the **Resource Control Plane**, where the SCA processes every received message and programs a given network equipment.

A. SLICE CONTROL AGENT ARCHITECTURE

The SCA consists of three main modules, SCA-API, SCA-Core, and the Data Plane Abstraction Service (DPAS).

- **SCA-API:**
  This module is the access point from/to the Infrastructure Control Plane. It consumes intent-based messages

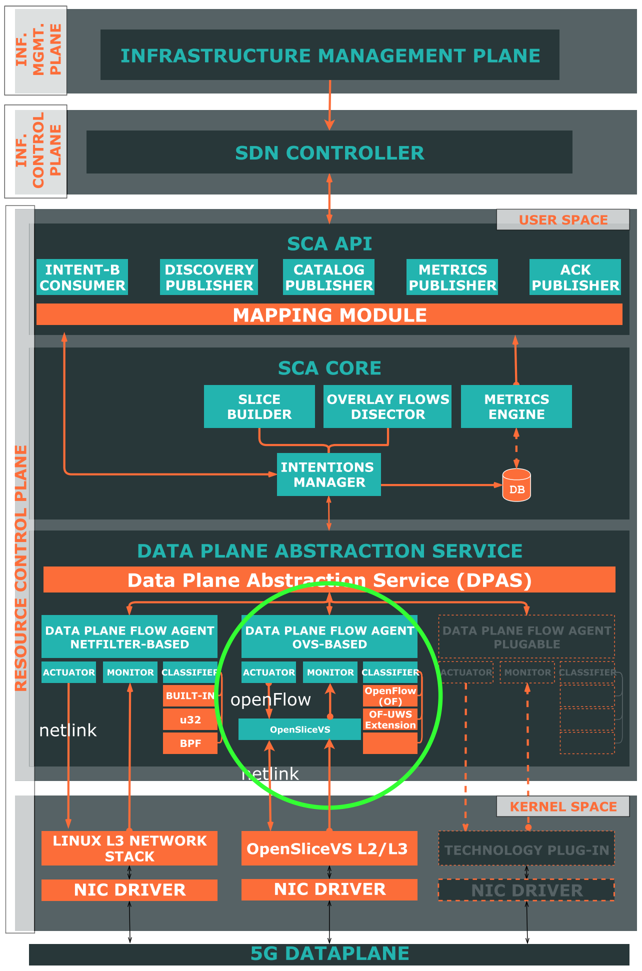


FIGURE 3. Network Slice Control framework architecture.

(further explained in Section IV-B) to apply actions at any location of the data path by exposing the NBI of the proposed service. When the SCA is instantiated, as a first step, it publishes its location and capabilities by using the discovery and catalogue sub-modules respectively. After that, it waits for new intent-based messages coming from its NBI. Additionally, the ACK sub-module publishes the result of any applied action over the data plane and the metrics sub-module provides behavioral information on the performance of a successfully applied slice.

- **SCA-Core:**
  The SCA-Core module processes incoming intent-based messages and manages the life cycle of the slices. It has the logic to dissect the packet representation structure into different sub-flows associated to each overlay network. The slice builder sub-module creates generic instructions to satisfy the slice requirements but delegates their implementation to other underlying modules that contain the logic for a specific technology. In addition, it also contains a metric sub-module in charge of calculating and sharing slice metrics by using its local database.

```
{
  "Resources": [{
        "resourceId": "F8A4C949",
        "encapsulationID1": "00000445",
        "encapsulationType1": "vxlan",
        "srcIP": "146.191.50.26",
        "dstPort": "4789",
},{
        "resourceId": "B6E6B2B3",
        "encapsulationID2": "8894D0D4",
        "encapsulationType2": "gtp",
    "srcIP": "10.100.0.19",
    "dstPort": "2152",
},{
        "resourceId": "5A07C580",
        "encapsulationID2": "8894D0D4",
        "encapsulationType2": "gtp",
        "srcIP": "192.188.0.140",
        "dstPort": "5004",
  }],
  "Intent": {
    "flowAgentName": "OVS-based"
    "actionType": "INSERT",
    "actionName": "CREATE_SLICE",
    "slice_id": "03E8",
    "priority": "1",
    "MAB" : "12000000",
    "MGB" : "10000000"
 },
  "Params": [{
    "paramName":  "interfaceName",
    "paramValue": "eth0"
    }]
}
```

Listing. 1. Intent-based Message.

- **Data Plane Abstraction Service (DPAS):**
  The DPAS is a module that provides an abstraction layer to deal with the different network traffic control technologies (Flow Agents) registered as plug-ins. Through this approach, the SCA-Core does not need to know how each of these Flow Agents works, thereby achieving a modular approach where the DPAS communicates within a common language with upper layers of the architecture and thus in a technology-dependent manner with any plug-able underlying Flow Agent. The aim of this module is to select a proper Flow Agent able to apply slicing policy rules in a specific hooking point (data path point) of the host computer.

B. INTENT DEFINITION FOR 5G IoT NETWORK SLICES

The Intent-based Consumer sub-module, depicted in the upper left corner of the resources control plane of Fig. 3, conceptualizes the lower level details used to interface with higher level infrastructure control components and decomposes concepts in the technical details to enable following sub-modules to program a target networking device on the data plane. Messages reaching this northbound interface are composed of one or more **resources** describing the flows to take action to, one **intent** that defines which kind of action has to be applied over such flows (e.g. create a slice), and a list of **parameters** for fine-grained specifications (e.g, target network interface, mode, time, duration, etc.).

Listing 1 gives an example of a message that could be received at the NBI of the SCA to be processed. This particular example shows an intent-based message with three resources, the original network traffic flow plus two encapsulations. Each resource contains valuable information about its own/previous encapsulation details and some networking parameters with specific protocol fields/values that represent the network traffic belonging to what has been defined as a slice. To keep this example simple for brevity, Listing 1 just includes *srcIP* and *dstPort* but any other field of any other protocol (if present in the packet structure) would also be allowed here. Moreover, the *Intent* section indicates to create a new slice where all network traffic matching this specification will be forwarded by using a *OVS-based* Flow Agent. Three fields of the *Intent* define the QoS requirements of the slice: *Priority*, *Minimum Guaranteed Bandwidth* (MGB) and *Maximum Allowed Bandwidth* (MAB) and they are further explained in subsection V-C. Finally, the section *Params* gives instruction to program this just over the interface *eth0*. It should be noted that Listing 1 just shows an example with the aim of simplifying the understanding of the structure of an Intent-based message, meaning that much more complex resources, type of actions, and fine-grain parameters are also allowed here.

C. SLICE CONTROL AGENT SOUTHBOUND: FLOW AGENT CAPABILITIES TO ENABLE NETWORK SLICING IN 5G IoT NETWORKS AND OPEN vSwitch EXTENSIONS

5G architectures are based on softwarization and virtualization, using SDN and NFV paradigms as enabling technologies [1], [23]. This allows different tenants to share the same physical infrastructure and, as a result, Telecommunication Service Providers (TSP) can reduce both Operational Expenditure (OPEX) and Capital Expenditure (CAPEX) [24]. Furthermore, each mobile IoT device has a dedicated logical connection to the 5G User Plane Function (UPF) to maintain connectivity across different networks. Therefore, controlling tenant isolation and device mobility are two crucial aspects of 5G IoT networks. In typical implementation, device mobility is supported by encapsulating the original IoT traffic with GTP, and tenant isolation is provided by using other tunneling protocols such as VXLAN, GRE, STT or GENEVE.

Fig. 4 gives an example of double encapsulated 5G IoT traffic where VXLAN has been chosen to create a virtual overlay network for each tenant over IoT traffic previously encapsulated with GTP. The GTP encapsulation and decapsulation are performed by the gNB in the RAN segment and the UPF in the core segment. Regarding tenant isolation, the virtual switches that interconnect the virtual machines in both edge and core segments are responsible for the encapsulation and decapsulation of the overlay traffic.

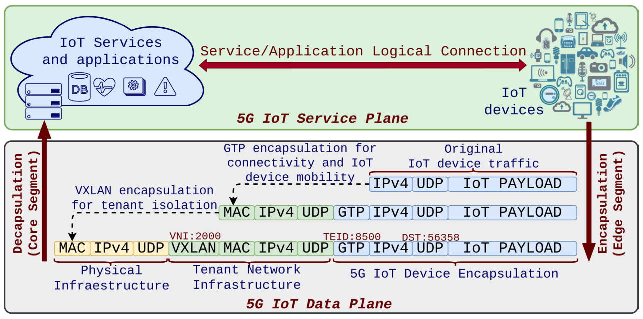


FIGURE 4. Double encapsulation in 5G networks for tenant isolation and 5G IoT traffic mobility.

This work adopts the definition of network slice proposed in [25]. The scope of this network slice definition is flexible enough to deal with all types of traffic that can be identified in a 5G network (such as those shown in Fig. 4) in order to meet the diverse QoS requirements that vertical use cases demand. In accordance with this definition, different slices can involve traffic with different levels of granularity depending on the use case addressed. For instance, a slice could be all the traffic of a particular tenant, the traffic of a vertical or, with a more fine-grained detail, the traffic of a single IoT device or even the traffic of a single service of a specific IoT device. In the experiments conducted to validate the proposed framework, a slice corresponds to the network traffic of a vertical, that is, the aggregate traffic transmitted by all the IoT devices of such specific vertical. Nevertheless, It is worth highlighting that our proposal is able to cover the wide scope of the slice definition proposed in [25].

Therefore, to provide fine-grained network slicing capabilities, the first stage is that the Flow Agents that handle network traffic in the data plane must be able to access the inner headers of the packets and extract information about the IoT devices including Source and Destination IP addresses, Source and Destination Ports, etc. and their associated metadata including GTP Tunnel Endpoint Identifier (TEID) and VXLAN network identifier (VNI). This scenario is exacerbated since 5G networks are complex heterogeneous ecosystems where different technologies co-exist and network traffic is treated by many different Flow Agents along the data path, both hardware and software, such as virtual switches, NICs, FPGAs or physical routers and switches.

As mentioned in subsection IV-A, the DPAS module of the proposed SCA mitigates this problem by providing the upper management layers with a common technology-agnostic API in a transparent manner regardless of the network device playing the role of Flow Agent and the underlying networking technology used. In addition, Flow Agents are required to have capabilities to enforce in the 5G data plane the QoS policies imposed by the upper management and control layer through the DPAS API.

The results presented in this paper are focused on a specific segment of the 5G data plane, the software data path, which is the segment that provides connectivity between the different VMs of the tenants allocated in the physical hosts of the 5G infrastructure. The Flow Agent deployed in the software data path of the proposed architecture is named *Open Slice Virtual Switch* (OpenSliceVS) and it has been implemented by making significant extensions to the base version 2.9.2 of OVS [26], [27]. OVS is an open source multi-layer virtual switch widely used in virtualized environments where

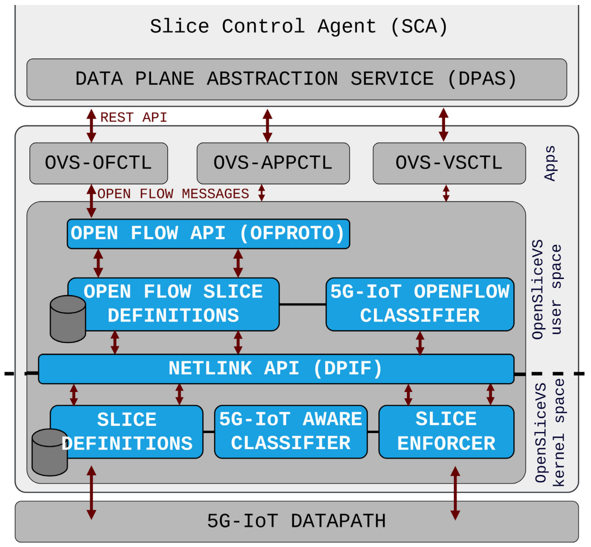


FIGURE 5. **Overview of the architecture of the OpenSliceVS software data path Flow Agent.**

SDN/NFV technologies play a key role [1], [23]. It provides OpenFlow capabilities to the SDN controller through the northbound interface. Since Linux kernel version 3.3, OVS kernel modules are available within the upstream Linux kernel distributions. OVS has become essential in SDN/NFV deployments and it is used in leading open source SDN/NFV oriented projects such as OpenDayLight [28] and OpenStack [29], among others. The following subsection provides an overview of the OpenSliceVS architecture and the extensions implemented to provide support for flexible, fine-grained control of 5G IoT traffic and network slicing capabilities in the software data path of the 5G IoT network.

D. OpenSliceVS FLOW AGENT: ARCHITECTURE AND IMPLEMENTATION DETAILS

The proposed prototype has been implemented by extending significantly several OVS features and functionalities in different modules throughout the OVS architecture, both in user and kernel spaces: traffic parser and classifier, flow and actions table, netlink communications between kernel module and user space daemon (DPIF), OpenFlow protocol, OpenFlow northbound API (OFPROTO) and command line applications interfaces. Fig. 5 shows an overview of the OpenSliceVS architecture, depicting the main modules in both user and kernel spaces. These are the main extensions implemented to the base version of OVS to provide network slicing support within the software datapath:

- **5G IoT aware Traffic classifier:** OVS delegates to the Linux kernel flow dissector module the extraction of the flow key that identifies a flow unequivocally. However, the Linux kernel does not support functionality to inspect protocol headers such as encapsulation headers or the inner headers of overlay traffic with several encapsulations that are expected in 5G IoT networks. To solve this, the prototyped Flow Agent has been enhanced with a new classifier that performs a deep inspection of the network packets, providing support for encapsulation/tunneling and other protocols not supported in the base version of OVS such as GTP, VXLAN, GENEVE, GTP and GRE. This new classifier allows re-entrance between headers that enable an efficient deep inspection of the inner headers of the overlay networks, gathering data about the tenants, the IoT devices and the verticals so that a fine-grained identification of all types of traffic present in 5G IoT networks is possible.
- **Open Flow tables:** The OpenFlow tables have been augmented with new fields that increase their expressiveness, providing a flexible, fine-grained slice definition that fits the intend-based message received by the SCA from the management plane. These new fields enrich the semantics of the left part of an OpenFlow rule, including metadata about verticals, IoT devices and encapsulation/tunneling protocols.
- **Netlink protocol:** The inter-process communication between the OVS kernel module (*openvswitch.ko*) and the OVS user space daemon (*ovs-vswitchd*) is implemented using netlink sockets provided by the netlink Linux kernel interface. The netlink handler sub-modules (*DPIF*) and the user-defined netlink messages have been extended to provide support for 5G IoT slicing capabilities.
- **North Bound Interface:** OVS exposes an OpenFlow-based NBI, named *OFPROTO*, to establish communication with the upper management and control layers on the basis of SDN principles. The OpenFlow messages and the NBI functionality to send and receive such OpenFlow messages have been extended to address the new fields added to the OpenFlow tables.
- **Command line applications:** Finally, the command line application suite included in the OVS distribution has also been upgraded with new capabilities to allow the upper layers to manage slices in the 5G IoT software data path either by the command line or via a REST API.

To conclude this subsection, the integration between OpenSliceVS and the DPAS module (Southbound interface of the SCA) is achieved via a REST API that enforces the slice polices (slice definitions and QoS requirements) in the 5G data plane using the command line applications provided by OpenSliceVS.

E. DYNAMIC MANAGEMENT OF THE LIFE CYCLE OF A SLICE IN THE DATA PLANE

A network slice is a dynamic entity, hence it is required a management of its life cycle as it has been outlined by Standard Development Organizations (SDOs) such as 3GPP [30], IETF [31] or 5G Americas [32]. The proposed framework offers to the upper control and management layers, a novel mechanism to control the complete life cycle of a slice within the data plane aligned with the specifications developed by

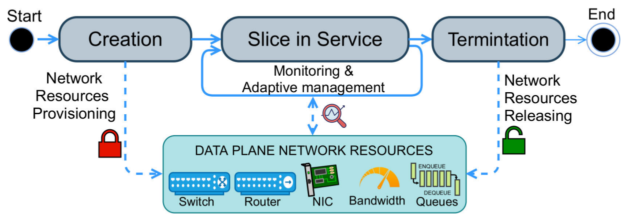

Start
Creation
Slice in Service
Termintation
End
Network Resources Provisioning
Monitoring & Adaptive management
Network Resources Releasing
DATA PLANE NETWORK RESOURCES
Switch
Router
NIC
Bandwidth
Queues
ENQUEUE
DEQUEUE

FIGURE 6. **Life cycle of a slice in the data plane. A slice can be in three different states: Creation, in Service and Termination.**

the SDOs. In this regard, a network slice instance can be in three different states as depicted in Fig. 6:

- **Creation:** Provision of the necessary network resources to ensure the performance isolation and QoS requirements of the slice.
- **In Service:** The slice has been successfully instantiated within the data plane and it is processing network traffic. In this state, the framework offers a dynamic change of the QoS parameters of the slice on request. This allows the network to adapt to the continuously evolving scenarios expected in 5G IoT use cases.
- **Termination:** Decommissioning of the network slice and releasing of the network resources attached to the network slice.

The sequence diagram provided in Fig. 7 details the logical steps involved in the process of creating a network slice, and how different components of the architecture cooperate to finally ensure the overall QoS of the data plane for a vertical service. As seen in Fig. 7, the sequence diagram is divided (horizontal gray lines) in three different stages: *SCA Instantiation*, *Slice Creation*, and *Vertical Use Case*. Each of these stages commences with an event, produced either by the management layer or the involved vertical, which triggers a sequence of automated steps.

- **SCA Instantiation (Steps 1-5):**
  First, after the infrastructure management has triggered a new SCA deployment, the SCA-Core requests for a list of available Flow Agents *(1)*. After that, the DPAS module provides such a list of Flow Agents, their capabilities, and their locations throughout the infrastructure *(2, 3)*. Finally, the SCA-Core sends this information, through its North Bound API *(4)*, to the external agent *(5)* so that the next stage can take place, which is the slice creation.
- **Slice Creation (Steps 6-15):**
  This stage starts when an external policy enforcer defines a new slice and sends an intent-based message for it to be enforced *(6, 7)*. Next, the SCA-Core processes the message *(8)* and delegates to the Data Plane Abstraction Service *(9)* the task of building a technology-dependant message *(10)*. At this point, the DPAS module selects a specific technology depending on the Flow Agent that is specified in the intent-based message, which in this example is OpenSliceVS (see Listing 1). In *(11)*, a tailored and technology-specific instruction is sent to this Flow Agent to create and enforce the requirements of the new slice in the data plane. In the OpenSliceVS example, the creation of a slice comprises two phases. First, the slice definition is inserted as an OpenFlow rule into the *slice definition and action table*. Second, a hierarchical token bucket (htb) queuing discipline configured with the QoS requisites of the new slice is created and attached to he network interface where the slice traffic will be redirected. Steps *12-15* are the way back up to the external SDN application so that it can be aware whether the network policy (the slice) has been applied.
- **Vertical Use Case (Steps 16-22):**
  This last stage begins when the service starts to be used *(16)* and its network traffic reaches a data-path point that has been programmed to satisfy the slice specific requirements. When a packet arrives at the Flow Agent, it is deeply inspected to extract data from the encapsulated inner headers *(17 and 18)*. This data, named *flow key*, includes, among other details, information about the vertical ID, the source IoT device and the destination IoT service, which is essential to provide a fine-grained slice control. The *flow key* is then compared with the *slice definition and action table* (19 and 20). If the search is successful, the packet is forwarded to the assigned slice, i.e., to the htb queuing discipline responsible for enforcing the QoS requirements of the slice *(21 and 22)*. On the other hand, if the *flow key* does not match any defined slice, a default policy (drop, sent to management plane, etc.) is applied.

The interaction between the different architectural components to update or delete a slice instance is very similar to the one described in steps 6 to 15 of the sequence diagram in Fig. 7 and therefore, for simplicity, no sequence diagram is included. The major difference lies in the fact that, to update or delete a slice instance, the SCA does not have to process a large and complex intend-based message (steps 8 and 9 in Fig. 7). Instead, the SCA delegates the task to the Flow Agent by passing the slice ID that uniquely identifies the slice throughout the data plane. Subsection VII-C provides an empirical analysis of the response time to handle a network slice instance in any of its states.

F. ALIGNMENT AND INTEGRATION WITH 5G INDUSTRY STANDARDS AND SPECIFICATIONS

To conclude this section, it worth to highlight that the solution proposed in this work is fully compliant and aligned with different standards and specifications developed by 5G industry and SDOs. Most of the efforts made by 3GPP are aimed at the design of a global standard for the 5G New Radio (5G NR), the air interface of 5G networks [33]. In this regard, our solution is deployed in the edge and core segments of the 5G network providing slicing capabilities across such network segments. Thus, it is complementary to 3GPP achieving an UEs end-to-end network slicing service. In relation to the

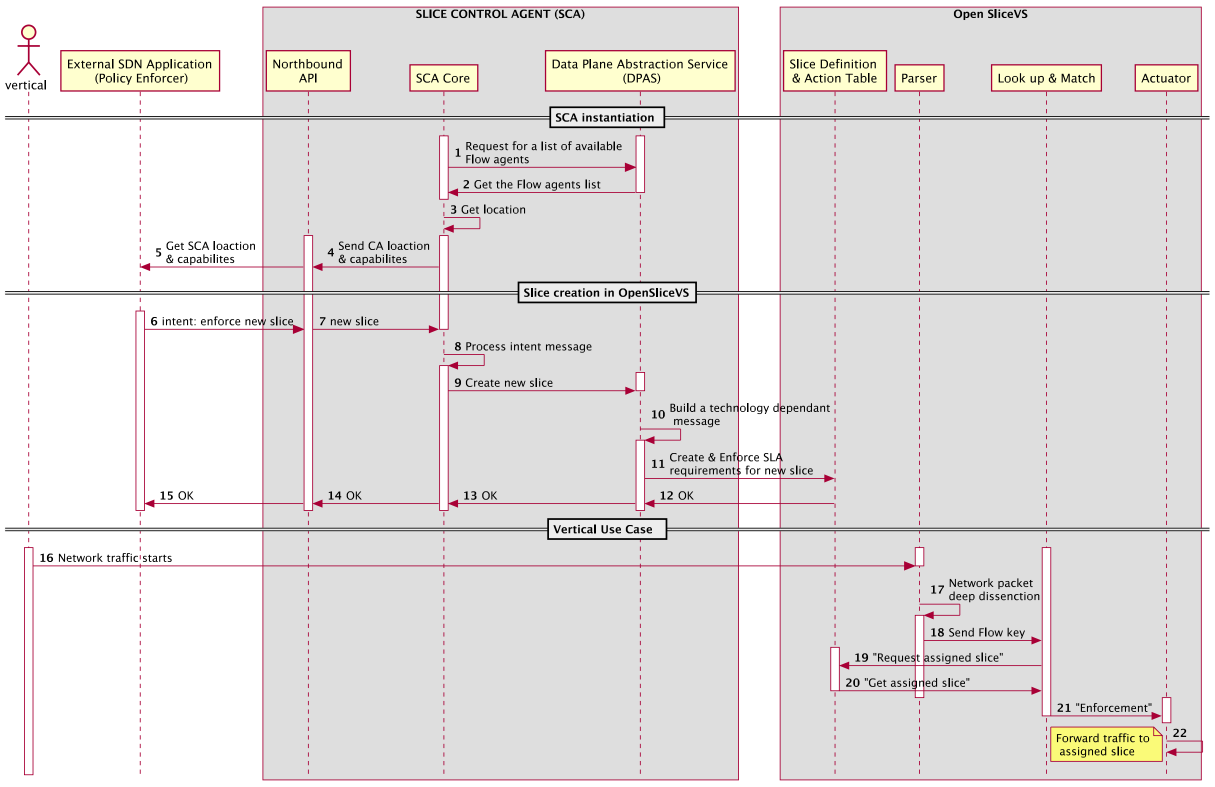

SLICE CONTROL AGENT (SCA)
Open SliceVS
vertical
External SDN Application (Policy Enforcer)
Northbound API
SCA Core
Data Plane Abstraction Service (DPAS)
Slice Definition & Action Table
Parser
Look up & Match
Actuator
SCA instantiation
1 Request for a list of available Flow agents
2 Get the Flow agents list
3 Get location
4 Send CA loaction & capabilites
5 Get SCA loaction & capabilites
Slice creation in OpenSliceVS
6 intent: enforce new slice
7 new slice
8 Process intent message
9 Create new slice
10 Build a technology dependant message
11 Create & Enforce SLA requirements for new slice
12 OK
13 OK
14 OK
15 OK
Vertical Use Case
16 Network traffic starts
17 Network packet deep dissenction
18 Send Flow key
19 "Request assigned slice"
20 "Get assigned slice"
21 "Enforcement"
22
Forward traffic to assigned slice

FIGURE 7. Sequence diagram describing the slice creation process and showing interaction between the actors involved: verticals, management plane, the SCA and the Flow Agent in the software data path (OpenSliceVS).

5G architecture proposed by ETSI which is oriented towards orchestration and virtualization aspects [34], our *SCA* module described above matches the functional description required for the component named *Network Slice Selection Function* (NSSF) in the ETSI specification. The main functionality of the NSSF is the selection of Data Plane network slices, which are created and configured with tailored QoS that fulfils the requirements of network users [35].

V. VERTICAL-ORIENTED IoT USE CASES FOR BENCHMARKING

To validate and demonstrate the suitability of the proposed framework, this paper considers five realistic IoT use cases that allow to evaluate the system's performance in scenarios where different verticals pose diverse requirements in terms of latency, data rate, reliability and scalability (number of IoT devices and number of slices). The characteristics and requirements of these use cases are listed in Table 1. As Fig. 8 illustrates, these use cases have been carefully chosen to cover the three general categories of services defined by ITU: eMBB, mMTC and URLLC [36], [37].

A brief overview of these three categories of services for 5G IoT use cases can be given as follows:

- **eMBB**: It supports enhanced 4G broadband traffic characterized by large payloads, moderate reliability and high data transmission rates. A moderate number of IoT devices are connected to the same base station (BS) demanding stable communication over time.

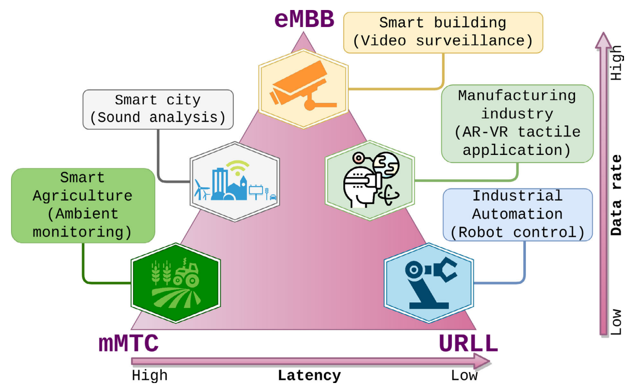

eMBB
Smart building (Video surveillance)
Smart city (Sound analysis)
Manufacturing industry (AR-VR tactile application)
Smart Agriculture (Ambient monitoring)
Industrial Automation (Robot control)
mMTC
URLL
High
Latency
Low
High
Data rate
Low

FIGURE 8. ITU category for every vertical oriented IoT use cases for evaluation testbed.

- **mMTC**: It supports a large number of IoT devices connected to the same cell, intermittently active and sending traffic with small payload at a low uplink transmission rate.
- **URLLC**: It supports traffic that demands very high reliability and low latency as main requirements, with a relatively low transmission rate of small payloads.

A. OVERVIEW OF THE VERTICAL-ORIENTED IoT USE CASES

Below we provide a description of every use case from a vertical point of view, underlining the QoS requirements

TABLE 1. Vertical oriented use cases: characteristics, KPIs and traffic profile.

	Use case 1	Use case 2	Use case 3	Use case 4	Use case 5
Use case	Ambient monitoring in Smart Agriculture	Robot control in Industrial Automation	Sound analysis in Smart Cities	Video surveillance in Smart Buildings	AV-VR tactile application in Manufacturing Industry
IoT Vertical role	Smart farm	Smart factory	Smart City	Smart Building	Smart manufacturing factory
Category	mMTC	URLLC	mMTC/eMBB	eMBB	URLLC/eMBB
Latency	High ($>$1s)	Low ($<$1 ms)	High ($>$100 ms)	High ($>$100 ms)	Low ($<$1 ms)
Reliability	Low	High	Medium	Medium	High
Priority	Low	High	Medium	Medium	High
Maximum Number of Verticals (Slices)	1000	100	24	40	24
Maximum IoT devices per Vertical	1000	2000	3000	100	20
Maximum total devices	10^6	$2\text{x}10^5$	72000	4000	480
Packet size transmitted by IoT device (Bytes)	128	256	1396	1396	1396
Double encapsulated packet size (Bytes)	232	360	1500	1500	1500
Packets per Second (PPS) per device	1	4	16	400	3300
Device Tx bandwidth	1.86 kbps	11.52 kbps	192 kbps	4.8 Mbps	39.6 Mbps
Maximum accumulated bandwidth	1.86 Gbps	2.40 Gbps	13.82 Gbps	19.20 Gbps	19.00 Gbps

demanded by verticals in real-world operations and the traffic profiles generated by the IoT devices, which are then emulated in our testbed:

- **Use case 1 (Ambient Monitoring in Smart Agriculture):** Smart agriculture aims to provide crop development optimization by means of environmental monitoring services. It provides farmers with timely information for better management and exploitation of their agricultural facilities, leading to maximized productivity. Each vertical deploys a large number of IoT sensors in their farms to perform real-time monitoring and gather a range of climatic metrics such as relative humidity, air temperature, soil temperature, light, soil moisture, wind or rain detection, to mention o few [38]. The traffic profile is typically packets with small payload sent every few seconds where requirements in terms of reliability, latency and throughput are not a big concern. Since it is a mMTC use case, one of the major challenges is to provide connectivity for a huge number of low-power wide-area (LPWA) IoT devices sending traffic intermittently. The proposed 5G IoT framework has been tested in providing service to up to 1000 farms (verticals) where each farm deploys up to 1000 IoT sensors. Therefore, the 5G IoT network processes traffic from up to 1 million IoT sensors simultaneously connected.
- **Use case 2 (Robot Control in Industrial Automation):** In Industrial Internet of Things (IIoT), often referred to as industry 4.0, robot automation processes are used to automate highly repetitive routine tasks traditionally performed by humans. Such tasks are now accomplished with high accuracy by robots in a faster and more efficient manner, eliminating errors introduced by human factors. This leads to a productivity boost and cost savings in time and money [39]. To test the proposed framework, we consider a smart manufacturing scenario where every vertical is defined as a smart factory using sensors and robots for material handling (picking and packing) and assembly tasks. NFV-based services located at the edge of the 5G network enable the automation of complex operations such as control and monitoring of the manufacturing process, resource and asset management or predictive maintenance. A system malfunction can have a negative impact on the overall network and therefore low latency and high reliability are crucial QoS requirements in this scenario. Such QoS requisites are characteristic of uRLLC use cases. In this use case, 100 smart factories are simultaneously connected to the 5G IoT network. An infrastructure made up of 2000 IoT devices (sensors and robots) has been deployed in every manufacturing plant whereby each IoT device sends traffic at a moderate constant throughput of 11.5 Kbps.
- **Use case 3 (Sound Analysis in Smart city):** The adoption of IoT in urban environmental acoustics helps provide solutions for use cases such as noise control at public events (concerts, festivals,...), detection of sound patterns for crime prevention (shots, screams, breaking glasses), traffic status monitoring or audio surveillance by using keyword spotting. The information extracted from sound data gathered by the city-wide acoustic IoT network is further analyzed by speech recognition and sound analysis algorithms [40]. Local authorities can benefit from this information and improve the quality of life of citizens in a efficient and cost-effective fashion. The scenario for testing the proposed framework consists of 24 cities, i.e. verticals. For every vertical, an IoT infrastructure comprising 3000 sound sensors has been deployed throughout the city, sending real-time ambient sound to the application servers located on the edge of the 5G network. The sound is encoded in MP3 quality at a bitrate of 192 kbps and sent in 1500 byte packets. The traffic profile of this use case fits in a hybrid mMTC/eMBB service.
- **Use case 4 (Video Surveillance in Smart Building):** Video surveillance in smart building combined with video analytics solutions is able to serve a wide range of applications such as crowd control, visitor management, lighting and HVAC control for efficient energy consumption or safety and fire prevention. Traditional video

cameras are replaced by IP devices that send data-rich video to application servers that analyze the data and trigger the appropriate actions. In this use case, IP video cameras send full HD 1080p H.264 encoded video with a standard frame rate (30 fps), which means a constant video bitrate of 5 Mbps per IoT device (recommended configurations for video bitrate and resolution settings in [41]). The buildings play the role of verticals and the proposed 5G IoT architecture provides connectivity for up to 40 buildings with up to 100 cameras deployed in each building. The traffic pattern corresponds to a moderate amount of devices forwarding large payload packets at a moderate rate, which is eMBB traffic.

- **Use case 5 (AR-VR Tactile Application in Manufacturing Industry):** In a smart manufacturing context, augmented reality (AR) focuses on improving the interaction between humans and machines by allowing humans to access digital information through a virtual layer on top of the physical world displayed on visualization devices such a smart glasses or tablets [42]. The applications are diverse: live guidance for maintenance, training for assembly or warehouse operations among many others. The use of AR in industrial environments involves real-time video processing that demands high bandwidth as well as transmission of critical information about real-time industrial processes that imposes extreme requirements in terms of reliability and latency. In our testbed, the 5G network provides coverage up to 24 verticals, i.e. 24 smart manufacturing factories. Each vertical is equipped with 20 AR units operating simultaneously in its premises requiring a bandwidth of 40 Mbps per AR unit. This is the most demanding of the five use cases, combining URLLC and eMBB features.

In a first set of experiments, in subsection VII-A, every use case has been empirically evaluated separately to provide an individualized analysis of the scalability and performance of the proposed solution when processing homogeneous traffic matching the traffic profile and QoS requirements demanded by the verticals for a specific use case.

Nevertheless, IoT ecosystems are fragmented by nature in multiple verticals with diverse requirements and, as a consequence, heterogeneous services are expected to coexist simultaneously in 5G IoT networks within the same physical infrastructure. Therefore, once the performance boundaries of each specific use case are known, in a second set of experiments, the proposed 5G IoT framework has been evaluated under heterogeneous traffic in a more complex and challenging scenario (see results in subsection VII-B).

B. BANDWIDTH OVERHEAD DUE TO DOUBLE ENCAPSULATION IN 5G IoT MULTI-TENANT NETWORKS

As mentioned in subsection IV-C, the double encapsulation of 5G- IoT network traffic is necessary to support tenant isolation and IoT devices mobility in multi-tenant 5G IoT virtualized infrastructures. On the other hand, the double encapsulation results in the network dealing with larger packets than the ones originally sent by the IoT devices. As a consequence, there is an overhead in the overall bandwidth that the 5G infrastructure has to process. The encapsulation process modifies the network packets by adding new headers at the beginning of the packets. The structure and size of these new headers are constant regardless of the size of the packets initially sent by the IoT devices. Hence, the smaller the packet size, the higher the proportional impact of the double encapsulation on the overall bandwidth and network resources needed to cope with the verticals' demands. In the experiments carried out in our testbed, VXLAN and GTP are the protocols employed to perform the double encapsulation and 104 extra bytes are added to every packet (as a result of inserting the sequence of headers *MAC-IPv4-UDP-VXLAN-MAC-IPv4-UDP-GTP* at the beginning of the packet).

It is noted that in Table 2 for use cases 1 and 2, which send packets with small payloads, the impact of the double encapsulation in the bandwidth is significant: 81% and 40% respectively. Regarding the remaining use cases (3, 4 and 5), the increase in the required bandwidth is lower, 7.44%. It is worth highlighting that all results given in this paper refer to double encapsulated traffic within the 5G infrastructure.

TABLE 2. Bandwidth overhead due to double encapsulation process in relation to the original packet size sent by the IoT devices.

Use case	Packet size Original / Encapsulated	Bandwidth overhead (Impact in %)
1	128 / 232	+ 81.25 %
2	256 / 360	+ 40.62 %
3,4,5	1396 / 1500	+ 7.44 %

C. PARAMETERS FOR QoS REQUIREMENTS OF A SLICE IN THE TESTBED

In terms of performance, the needed parameters to set up a slice are three: *Maximum allowed bandwidth (MAB)*, *minimum guaranteed bandwidth (MGB)* and *priority*. The MAB and the MGB values determine respectively the upper and lower boundaries within a vertical is allowed to send traffic according to its SLA. A traffic rate above the upper limit would be a SLA violation by the vertical and it should not be permitted by the control and management layer under any circumstances. A traffic rate lower than the minimum limit may indicate an under-utilization of the provisioned resources and a revision of the network slices configuration would be useful to optimize the available network resources.

For the experiments conducted in our testbed (see results in sections VII), the slices have been configured using the expected bandwidth as a reference value to determine the MAB and the MGB parameters, assigning an interval of ±25% of the expected bandwidth to every slice. Fig. 9 illustrates this with an example for use case 5. For this use case, every single IoT device transmits at a constant data rate of 39.6 Mbps. In the more complex scenario with 24 verticals and 20 IoT devices per vertical, the expected bandwidth of

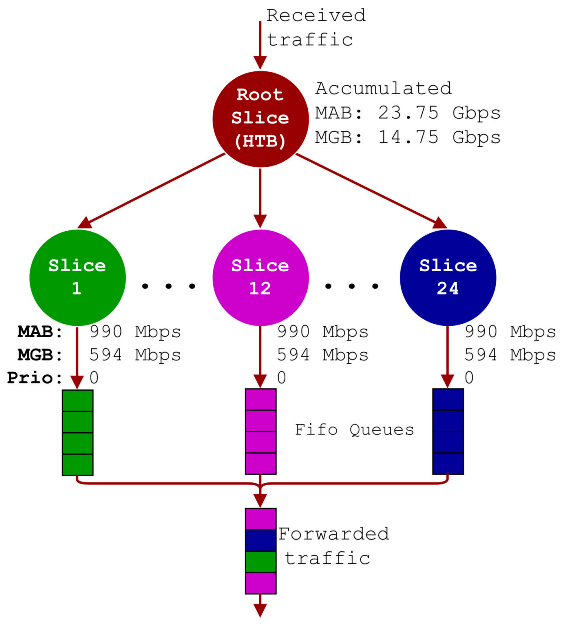


FIGURE 9. Maximum allowed bandwidth (MAB) and minimum guaranteed bandwidth (MGB) for use case 5 in a scenario with 24 smart manufacturing factories (slices) with 20 AR units per factory.

every vertical is 792 Mbps. The accumulated expected bandwidth, i.e., the sum of the bandwidth of all slices that the 5G IoT infrastructure must handle, is 19 Gbps. Thus, the MAB and MGB for every single slice are 990 and 594 Mbps respectively (792 Mbps ±25%). Similarly, the accumulated values of MAB and MGB are 23.75 Gbps and 14.75 Gbps respectively (19 Gbps ±25%).

Regarding priority, this parameter is tightly linked to reliability and latency. On every interface, the traffic scheduler prioritizes higher priority traffic over lower priority traffic. Consequently, packets with higher priority are queued for a shorter time and are forwarded before packets with lower priority. The direct implications on high priority traffic when compared with lower priority traffic are higher reliability (a packet is less likely to be dropped due to a full queue) and lower delay (due to lower queuing delay). It is noted that priorities are set differently depending on whether the traffic is homogeneous or heterogeneous. In the experiments conducted with homogeneous traffic in subsection VII-A, since all network traffic corresponds to the same ITU category, all slices have been set with the same priority and, hence, equally treated. Regarding the experiments with heterogeneous traffic, the priority of every slice has been set according to the latency and reliability requisites demanded by the use case (see subsection VII-B).

VI. IMPLEMENTATION DETAILS AND TESTBED FOR EXPERIMENTATION

This section provides details about the testbed infrastructure deployed to empirically validate and evaluate the framework proposed, and the methodology adopted to execute the experiments and analyze the gathered results.

A. HARDWARE AND SOFTWARE SPECIFICATIONS FOR TESTBED

The experiments have been carried out on a computer with the following specifications: 2-node NUMA architecture where each node was an Intel Xeon CPU E5-2660 v4 based on Broadwell architecture with 14 cores operating at 2 GHz and hyper-threading features. The system had a 128 GByte RAM and a 2 Terabyte SCSI HDD. The operating system was CentOS release 7.8.2003 running a Linux kernel version 3.10.0.

Regarding the virtualisation software, the hypervisor was qemu version 1.5.3 using libvirt version 4.5.0. The CPU provides a constant Time Stamp Counter (TSC) that allows, using the virtual Precision Time Protocol (PTP) hardware clock provided by the Linux kernel, a time synchronization between the host and the guests with a sub-microsecond accuracy.[1] This is needed to get accurate time stamping of network traffic in virtualized environments like our testbed infrastructure where events such as interruptions, process migrations or deep C states could lead to issues of time keeping in guest VMs. Therefore, the empirical results achieved in this paper are accurate and trustworthy with an error of the order of nanoseconds.

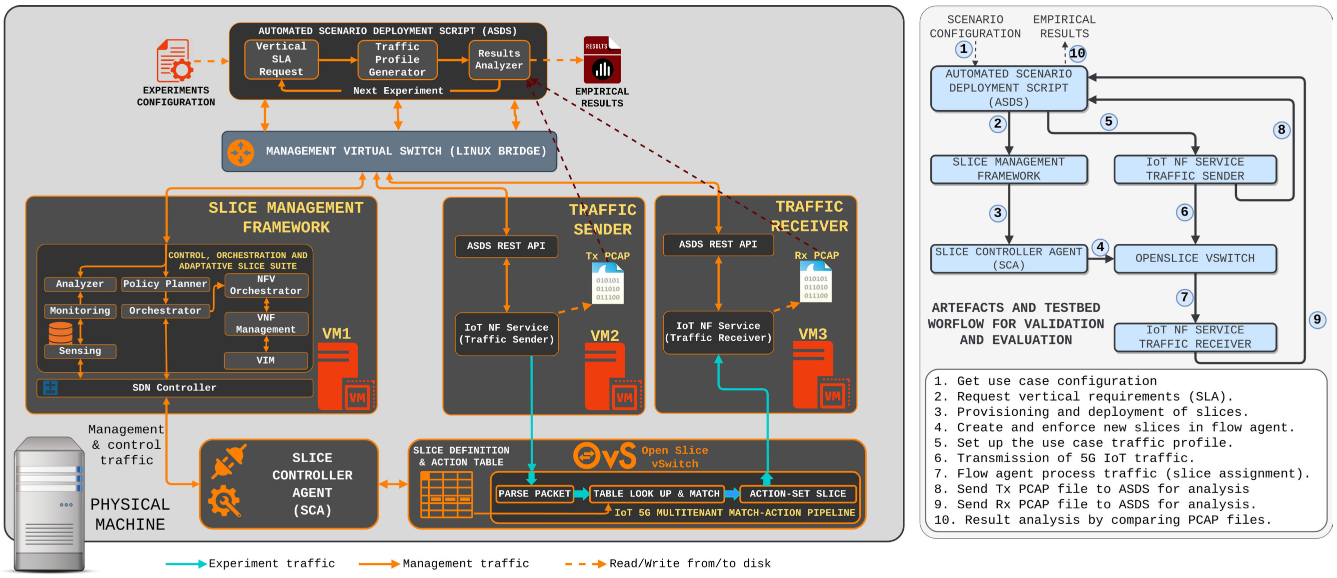
B. METHODOLOGY TO EXECUTE EXPERIMENTS AND ANALYZE RESULTS

Concerning the procedure followed to gather results, all the experiments have been reproduced for ten times. The results in the best one and the worst one have been discarded to prevent outliers and, therefore, the results (delay, packet loss, bandwidth, etc.) shown in this work are the arithmetic mean of the eight remaining intermediate values. For every single experiment, network traffic hes been sent for at least 30 seconds, mapping the use case traffic profile, the amount of IoT devices connected and the number of verticals. Based on previous test carried out, 30 seconds is a reasonable time interval to produce trustworthy and accurate results.

C. TESTBED INFRASTRUCTURE AND IMPLEMENTATION DETAILS

Fig. 10 illustrates the testbed deployed to carry out the empirical validation and evaluation of the 5G IoT network slicing capabilities of the novel proposed framework. This figure also shows the software artefacts in the testbed and the workflow to conduct each single experiment, starting when the vertical requests subscription to a network service until the data gathered during the experiment are analyzed to obtain empirical results. The testbed infrastructure consists of three virtual machines. The first VM (VM1) is responsible for a centralized management of the infrastructure, and thus accommodates the control and management modules described in subsection III-C3 and the SDN controller. VM2 and VM3 virtual machines host NF-based virtualized services

[1] KVM guest timing management: `https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/virtualization_deployment_and_administration_guide/chap-kvm_guest_timing_management`


AUTOMATED SCENARIO DEPLOYMENT SCRIPT (ASDS)
Vertical SLA Request
Traffic Profile Generator
Results Analyzer
Next Experiment
EXPERIMENTS CONFIGURATION
EMPIRICAL RESULTS
MANAGEMENT VIRTUAL SWITCH (LINUX BRIDGE)
SLICE MANAGEMENT FRAMEWORK
CONTROL, ORCHESTRATION AND ADAPTATIVE SLICE SUITE
Analyzer
Policy Planner
Monitoring
Orchestrator
Sensing
NFV Orchestrator
VNF Management
VIM
SDN Controller
VM1
TRAFFIC SENDER
ASDS REST API
Tx PCAP
IoT NF Service (Traffic Sender)
VM2
TRAFFIC RECEIVER
ASDS REST API
Rx PCAP
IoT NF Service (Traffic Receiver)
VM3
Management & control traffic
PHYSICAL MACHINE
SLICE CONTROLLER AGENT (SCA)
SLICE DEFINITION & ACTION TABLE
Open Slice vSwitch
PARSE PACKET
TABLE LOOK UP & MATCH
ACTION-SET SLICE
IoT 5G MULTITENANT MATCH-ACTION PIPELINE
Experiment traffic
Management traffic
Read/Write from/to disk
SCENARIO CONFIGURATION
EMPIRICAL RESULTS
AUTOMATED SCENARIO DEPLOYMENT SCRIPT (ASDS)
SLICE MANAGEMENT FRAMEWORK
IoT NF SERVICE TRAFFIC SENDER
SLICE CONTROLLER AGENT (SCA)
OPENSLICE VSWITCH
IoT NF SERVICE TRAFFIC RECEIVER
ARTEFACTS AND TESTBED WORFLOW FOR VALIDATION AND EVALUATION

1. Get use case configuration
2. Request vertical requirements (SLA).
3. Provisioning and deployment of slices.
4. Create and enforce new slices in flow agent.
5. Set up the use case traffic profile.
6. Transmission of 5G IoT traffic.
7. Flow agent process traffic (slice assignment).
8. Send Tx PCAP file to ASDS for analysis
9. Send Rx PCAP file to ASDS for analysis.
10. Result analysis by comparing PCAP files.

FIGURE 10. Architecture, artifacts and workflow of the testbed deployed for the empirical evaluation of our proposed 5G IoT Slice Management Framework.

that send and receive 5G IoT traffic respectively. Connectivity between VM2 and VM3 is provided by an instance of *OpenSliceVS*, the Flow Agent tasked with provisioning and enforcing slicing policies in the software data path segment of the data plane. Therefore, the results achieved in this work correspond to an Edge-to-cloud network slicing preformed between VM2 and VM3. The communication between the Slice Management suite located in VM1 and the *OpenSliceVS* instance is ensured in a technology-independent way by an SCA instance as described in subsection IV-A.

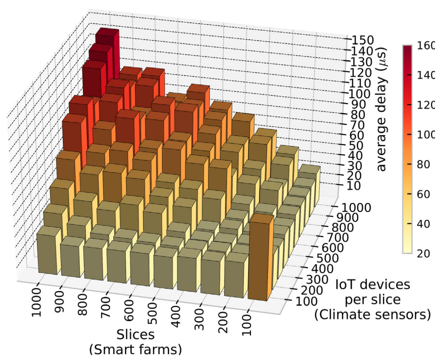
The experiments are conducted by the *Automated Scenario Deployment Script* (ASDS). This tool has been developed to evaluate in an automated manner the performance of the proposed 5G IoT Network Slicing framework in different scenarios. It is able to execute large batches of experiments whose parameters have been previously defined in a JSON file. The workflow to conduct an experiment is as follows: After reading the experiment configuration (see 1 in Fig. 10), the ASDS agent assumes the role of the verticals and sends a request to the *Slice Management Framework* about the SLAs corresponding to every vertical involved in the ongoing experiment (see 2 in Fig. 10). Then, the *Slice Management Framework* enforces the slicing policies derived from the SLAs subscribed by the vertical and provisions the needed resources to ensure the demanded network slicing service. Using the *Data Plane Abstraction Service* (DPAS) provided by the SCA, the slices are created in the data plane by the software data path Flow Agent (*OpenSliceVS*) (see 3 and 4 in Fig. 10). At this point, the 5G infrastructure is aware of the verticals' requirements, and the necessary hardware and software resources to meet them have been provisioned. Hence, the verticals' IoT devices can start to send network traffic to the virtualized IoT services allocated in the servers of the 5G network. Next, the ASDS agent sends to the IoT service in VM2 the traffic profile of the experiment (including the number of IoT devices, transmission bandwidth per device, payload size, etc.) (see 5 in Fig. 10). The NF service allocated in VM2 generates network traffic matching this pattern. This traffic is sent to the *OpenSliceVS* instance where the slicing is performed and the packets are forwarded to the IoT NF service allocated in VM3 (steps 6 and 7 in Fig. 10). Every single packet is uniquely tagged along the experiment by a unique 8-byte ID and its associated slice ID, both included in the payload. Once the transmission has been conducted, both IoT services, sender and receiver, generate a PCAP file that is reported to the ASDS (8 and 9 in Fig. 10). Finally, the ASDS agent compares the data of both PCAP files (timestamp, packet ID and associated slice ID) to obtain the empirical results of the experiment.

VII. EMPIRICAL VALIDATION, EVALUATION AND ACHIEVED RESULTS

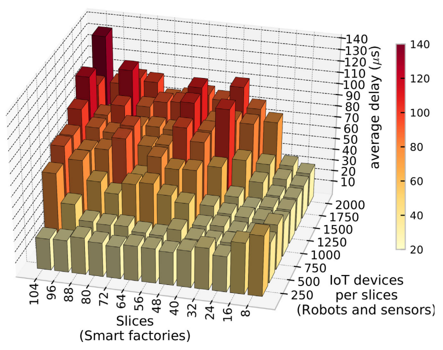
In this section, we first provide an empirical evaluation and analysis of the scalability of the achieved performance in terms of delivered QoS (latency, packet loss and bandwidth) when dealing with homogeneous 5G IoT traffic. Next, the proposed framework is evaluated in a more demanding scenario with heterogeneous 5G IoT traffic where we focus on results about priority QoS requirements and performance isolation between slices. Finally, we provide results about the flexibility and agility of the system when managing the life cycle of a slice (provisioning, modification and decommissioning of a slice).

A. HOMOGENEOUS SCALABILITY OF 5G IoT NETWORK SLICING

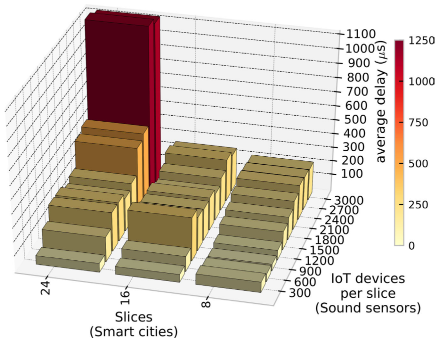
This subsection provides an empirical analysis of the proposed framework suitability when processing homogeneous 5G IoT traffic, i.e., network traffic from only one of the use cases introduced in Section V. Since all network traffic is of



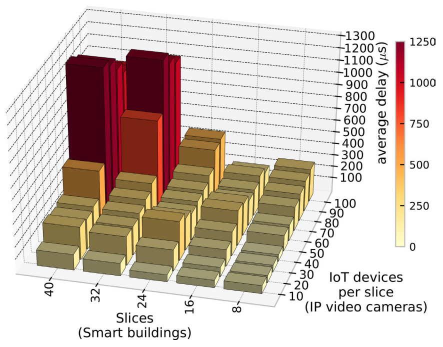
(a) Use case 1:
Ambient monitoring in Smart Agriculture.
Up to 1000 slices and 1,000,000 IoT devices sending 1,000,000 PPS at 1.86 Gbps.



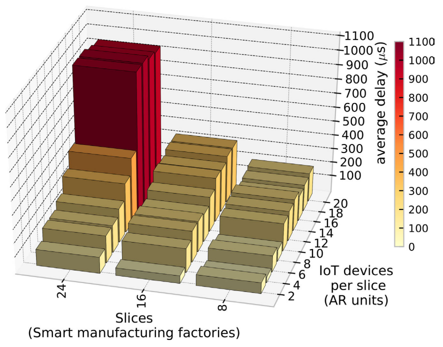
(b) Use case 2:
Robot control in Industrial Automation.
Up to 104 slices and 208,000 IoT devices sending 832,000 PPS at 2.40 Gbps.



(c) Use case 3:
Sound analysis in Smart City.
Up to 24 slices and 72,000 IoT devices sending 1,152,000 PPS at 13.82 Gbps.



(d) Use case 4:
Video surveillance in Smart Building.
Up to 40 slices and 4,000 IoT devices sending 1,600,000 PPS at 19.20 Gbps.



(e) Use case 5:
AV-VR tactile app. in Manufacturing Industry.
Up to 24 slices and 480 IoT devices sending 1,584,000 PPS at 19.00 Gbps.

FIGURE 11. Scalability results when processing homogeneous 5G IoT traffic (average end-to-end delay per packet in μs). For each use case, the framework has been evaluated ranging the number of slices (verticals) and the number of IoT devices per slice. It is given the maximum number of IoT devices per slice, packets per second (PPS) transmitted and accumulated Tx bandwidth configured for each use case.

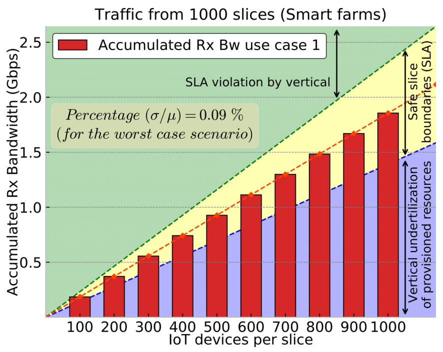
TABLE 3. Average delay per packet for the most stressful configuration for every use case.

Use case	Testbed configuration for the most congested scenario (Verticals and IoT devices per vertical)	Total IoT devices	Average delay (μs)
1	1,000 Smart Farms with 1,000 Climate Sensors	1,000,000	143
2	104 Smart Factories with 2000 Robots	208,000	133
3	24 Smart Cities with 3000 Sound Sensors	72,000	1054
4	40 Smart Buildings with 100 IP Cameras	4,000	955
5	24 Smart Factories with 20 AR Units	480	945

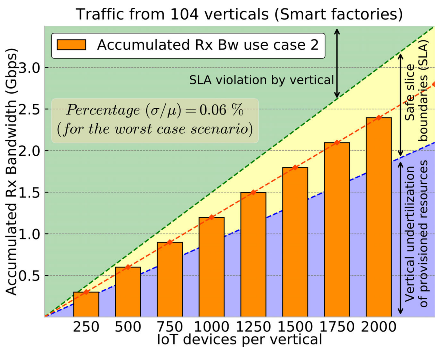
the same type, all slices are configured with the same priority and thus, equally treated with a round robin policy by the queuing scheduler.

Fig. 11 shows the system performance in terms of latency (average delay per packet) when ranging the number of verticals and the number of IoT devices per vertical for each use case. As it can be observed in Table 3, the latency requirements listed in Table 1 are fulfilled for all the use cases even in the most stressful scenarios (configurations with the highest number of verticals and 5G IoT devices). It is worth highlighting that, for the majority of the scenarios, the average delay is in the order of μs while for some of the most demanding scenarios it is slightly over 1 ms. Packet loss ratio is 0% for all the executed scenarios in all the use cases, and thus no graph is provided regarding packet loss.

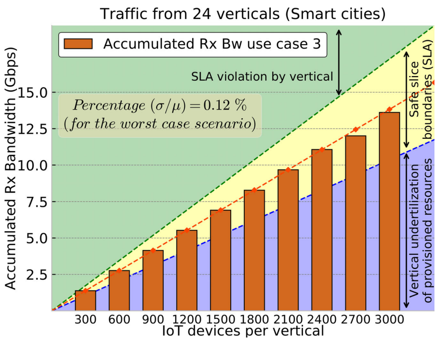
In use case 1 (mMTC traffic), the maximum measured delay is 143 μs when handling the emulated traffic from 1000 smart farms equipped with 1000 IoT climate sensors. Therefore, the proposed framework is able to cope with one million mMTC IoT devices with 0% packet loss and yield very low latency. These values are significantly lower than the requirements specified for this type of traffic in Table 1. Concerning use case 2, since it involves uRLLC traffic, high reliability and low latency are strict requirements that must be fulfilled. The outcome of the experiments carried out on our testbed proves that the proposed solution is able to cope with both successfully: low latency (133 μs for the worst case) and high reliability (0 % packet loss). Unlike use cases 1 and 2, latency and reliability are not challenging requirements for use cases 3 and 4. Notwithstanding, the gathered results are beyond the requirements imposed by the verticals: maximum average delay of around 1 ms (significantly less than the 100 ms specified for both use cases) and 0% packet loss even



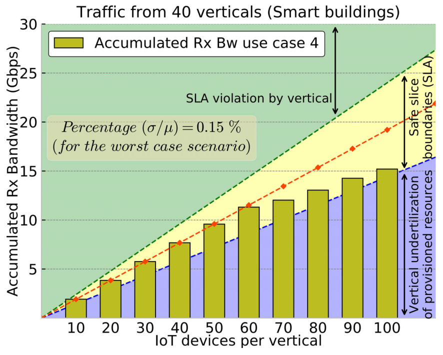
(a) Use case 1:
Ambient monitoring in Smart Agriculture.
1000 slices and up to 1,000,000 IoT devices
sending 1,000,000 PPS at 1.86 Gbps.



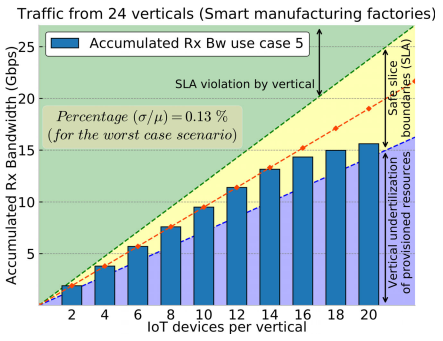
(b) Use case 2:
Robot control in Industrial Automation.
104 slices and up to and 208,000 IoT devices
sending 832,000 PPS at 2.40 Gbps.



(c) Use case 3:
Sound analysis in Smart City.
24 slices and and up to 72,000 IoT devices
sending 1,152,000 PPS at 13.82 Gbps.



(d) Use case 4:
Video surveillance in Smart Building.
40 slices and and up to 4,000 IoT devices sending
1,600,000 PPS at 19.20 Gbps.



(e) Use case 5:
AV-VR tactile app. in Manufacturing Industry.
24 slices and and up to 480 IoT devices sending
1,584,000 PPS at 19.00 Gbps.

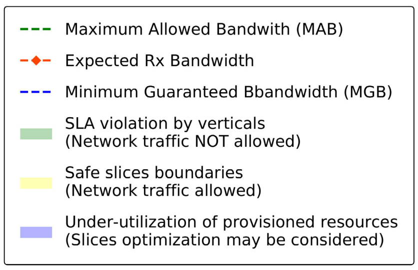

Maximum Allowed Bandwith (MAB)
Expected Rx Bandwidth
Minimum Guaranteed Bbandwidth (MGB)
SLA violation by verticals (Network traffic NOT allowed)
Safe slices boundaries (Network traffic allowed)
Under-utilization of provisioned resources (Slices optimization may be considered)

FIGURE 12. **Accumulated Rx Bandwidth (in Gbps) when processing homogeneous 5G IoT traffic and safe slice boundaries for each use case. It is given the maximum number of IoT devices per slice, packets per second (PPS) transmitted and accumulated Tx bandwidth configured for each use case.**

though both use cases are tolerant to packet loss. Finally, use case 5 is the one that demands more severe requirements since its traffic profile corresponds to a hybrid uRLLC/eMBB category. As with the previous use cases discussed, the proposed 5G IoT framework is able to guarantee the requirements imposed by the verticals of the use case 5: low latency (delay lower than 1 ms in the most stressful configuration with 24 verticals and 20 AR units per verticals) and high reliability (0% packet loss) whilst being able to process a high bandwidth inherent to eMBB traffic.

So far, the performance achieved in relation to latency and reliability has been analyzed. Regarding bandwidth requirements, Fig. 12 shows the accumulated bandwidth achieved for every use case. For clarity and simplicity, the results showed for every use case corresponds to the scenarios with the highest number of verticals (slices) when ranging the number of IoT devices connected to the 5G IoT network. The results gathered with fewer verticals are similar or even better due to the fact that these are configurations where the network is less congested. Three different areas can be observed in every graph. The central yellow area indicates the bandwidth interval for every slice. The upper green and lower blue dotted lines that delimit this area determine the values of the MAB and the MGB as described in subsection V-C. The orange dotted line in the centre is the accumulated expected bandwidth used as a reference to calculate the bandwidth limits of the slices. Consequently, no traffic should be allowed in the upper green area as it is a violation of the SLAs by the verticals. Finally, the verticals can send traffic bellow the MGB (the lower blue are in every graph), although, as mentioned in subsection V-C, it indicates an under-utilization of the provisioned resources for the slice.

The results shown in Fig. 12 demonstrate that the infrastructure is able to fulfill the bandwidth requirements for all the executed scenarios. In fact, for most configurations, the accumulated bandwidth measured matches pretty closely the expected one. Only for the scenarios with the higher number of IoT devices deployed in uses cases 4 an 5, the accumulated bandwidth achieved is bellow the expected one, but still within the guaranteed limits. Since the framework is dealing with homogeneous traffic where all the verticals (slices) demand the same requisites in terms of bandwidth, it is expected that the 5G IoT network will treat all the verticals in the same manner. In a first approach to illustrate this system

TABLE 4. Testbed setup for empirical evaluation with heterogeneous 5G IoT traffic. Use cases ordered by priority.

Use case	Testbed configuration (Verticals and IoT devices per vertical)	IoT devices	PPS	Bandwidth	Priority
2 Industrial Automation	8 Smart Factories with 2000 Robots	16,000	64,000	0.18 Gbps	2 (Highest)
5 Manufacturing Industry	8 Smart Factories with 15 AR Units	120	396,000	4.75 Gbps	3
3 Smart Cities	8 Smart Cities with 3000 Sound Sensors	24,000	384,000	4.61 Gbps	4
4 Smart Buildings	8 Smart Buildings with 100 IP Cameras	800	320,000	3.84 Gbps	5
1 Smart Agriculture	400 Smart Farms with 600 Climate Sensors	240,000	240,000	0.45 Gbps	6 (Lowest)
	432 Slices (1 slice per vertical)	280,920	1,404,000	13.83 Gbps	

behaviour, we intended to plot the standard deviation of the achieved bandwidth per slice with a whisker plot on top of each bar in the bar graphs in Fig. 12. However, the standard deviation values calculated are so close to the arithmetic mean that, once plotted, they are imperceptible in the bar graphs. In a second approach to assess the system's behaviour in relation to this, we provide the ratio (in percentage) between the standard deviation (σ) of the achieved bandwidth per slice and its arithmetic mean (μ) for the worst execution of every use case (see annotations in green boxes for every use case in Fig. 12). This ratio is a good indicator of how equally slices are treated by the network in terms on bandwidth. From a statistical viewpoint, 95% of the values for the achieved bandwidth per slice are in the interval [$\mu - 2\sigma, \mu + 2\sigma$]. Therefore, the smaller this interval is, the more equally the slices are treated in terms of achieved bandwidth. The highest value of this indicator is 0.13% (use case 3), which indicates a good stability and robustness of the system. Even in the more congested scenarios where the accumulated bandwidth measured is below the expected one, the available bandwidth is distributed almost equally among all the slices with the same priority and throughout requisites (see executions with higher number of IoT devices in use case 4 and 5).

To conclude this subsection, the gathered results demonstrate that the proposed framework is able to meet the verticals requirements in terms of latency, reliability (packet loss) and throughput when dealing with homogeneous traffic where all slices match traffic with the same profile and requirements. The proposed solution provides good scalability while being flexible enough to adapt to the diverse requirements of the use cases analyzed in this work: it is able to provide connectivity for up to 1 million IoT devices in mMTC traffic, achieve over 15 Gbps bandwidth in congested eMBB scenarios or ensure delays in the order of μs for critical-missions communications (uRLLC), providing high reliability in all of the tested scenarios (0% packet loss ratio).

B. HETEROGENEOUS SCALABILITY OF 5G IoT NETWORK SLICING

The prototyped 5G IoT framework has also been evaluated in a more challenging scenario where traffic from the 5 use cases demanding diverse QoS requirements is competing simultaneously for the available network resources. Table 4 provides a breakdown of the heterogeneous traffic profile of the experiments conducted. The configuration of

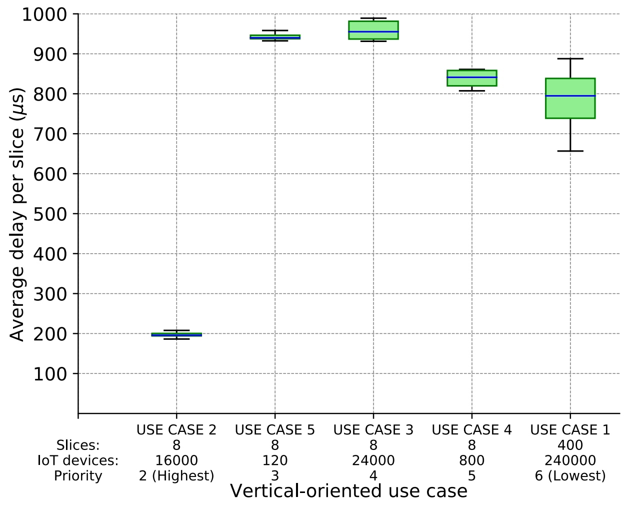


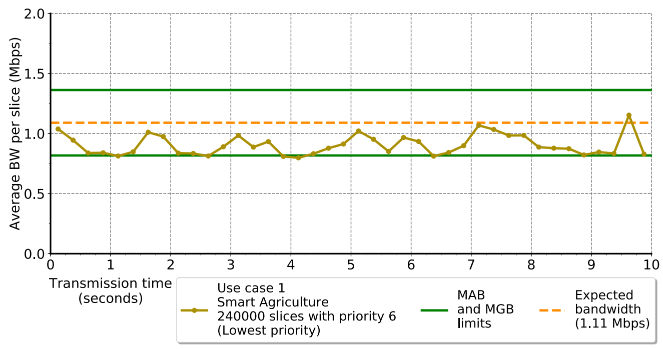
FIGURE 13. Average delay per slice and use case (in μs) when processing heterogeneous 5G IoT from 280,920 IoT devices and 432 slices enforced in the software data plane by the OpenSliceVS Flow Agent.

TABLE 5. Example of proposed priorities mapping different 5G IoT traffic profiles.

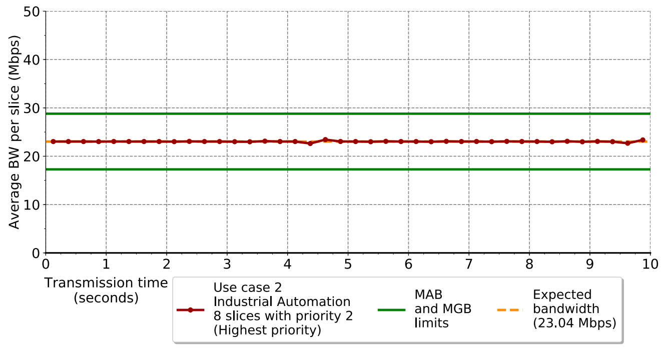
Type of 5G traffic	Slice priority
Network control and management	0 (Highest)
Critical communications	1
Delay-sensitive and high reliable URLLC traffic	2
Delay-sensitive and high reliable bulk URLLC/eMBB traffic	3
High latency mMTC/eMBB traffic	4
High latency eMBB traffic	5
High latency and packet loss tolerant mMTC traffic	6
Best effort	7 (Lowest)

this scenario is a realistic example of 5G IoT traffic which is expected in a real-world 5G IoT infrastructure. As it can been noticed, the proposed framework has been tested with 432 slices sending traffic from a total of 280,920 IoT devices. In terms of bandwidth, the 5G network handles 1,404,000 packets per second (PPS), reaching a combined bandwidth of 13.83 Gbps whereas it has to ensure the verticals' QoS requirements committed in the SLAs.

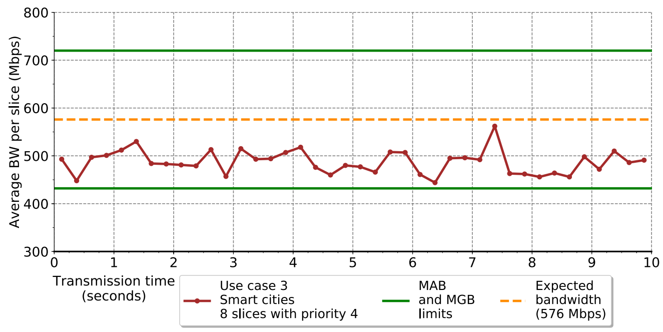
Unlike the previous experiments with homogeneous traffic, priority is a key QoS parameter when dealing with heterogeneous 5G IoT traffic in a more complex scenario where it is required to guarantee delay-sensitive and high-reliable services. In congested networks with high traffic load, packets are buffered before being forwarded, causing an additional delay. Moreover, if the buffer becomes full, packets



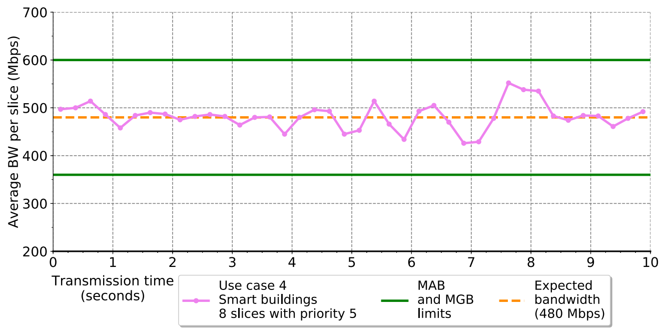
(a) Use case 1: Ambient monitoring in Smart Agriculture
400 Smart Farms (slices) with 600 IoT sensors deployed in each farm, transmitting a total of 240,000 PPS.
Total BW achieved: 0.360 Gbps.



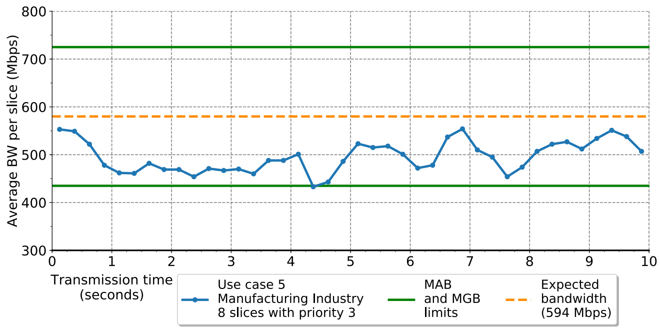
(b) Use case 2: Robot Control in Industrial Automation
8 Smart Factories (slices) with 2000 robots per factory, transmitting a total of 64,000 PPS.
Total BW achieved: 0.184 Gbps.



(c) Use case 3: Sound analysis in Smart City
8 Smart cities with 3000 IoT microphones distributed alongside each city, transmitting a total of 384,000 PPS.
Total BW achieved: 3.986 Gbps.



(d) Use case 4: Video surveillance in Smart Buildings
8 Smart buildings, each one equipped with 100 IP video cameras, transmitting a total of 320,000 PPS
Total BW achieved: 3.848 Gbps.



(e) Use case 5: AR in manufacturing industry
8 Smart factories and 15 AR units operating in each one, transmitting a total of 396,000 PPS.
Total BW achieved: 3.960 Gbps.

FIGURE 14. Average bandwidth per slice for each use case in a scenario with heterogeneous 5G IoT traffic. The measured results correspond to a 10-second interval where 5G network traffic is simultaneously sent for 5 vertical-oriented use cases. The combined total bandwidth achieved is 12.36 Gbps.

are dropped. This fact might compromise the fulfillment of the QoS parameters specified in the SLA in terms of latency and reliability. Prioritization mechanisms address this issue by forwarding high priority traffic (demanding low latency and/or high reliability) before other traffic with less strict QoS needs. Table 5 provides a proposal on assigning priorities to different types of 5G network traffic. These values have been used to set the priority for the traffic of each vertical business in the experiments carried out in the testbed (see the fifth column in Table 4).

Fig. 13 displays the average delay per slice for every use case when the 5G infrastructure processes traffic from all 5 use cases simultaneously. The data correspond to network traffic captured for 10 seconds in the testbed. There are several conclusions that can be drawn from this graph. Firstly, an average delay below 1 ms is achieved for all slices. Secondly, as expected, the traffic transmitted by the IoT devices of the use case with the highest priority (URLLC category) gets the shortest delay (around 200 μs). Furthermore, it can been observed that the slices of use cases with higher priority

(2 and 5) obtain a rather stable delay in comparison with the remaining use cases (the whisker box's height is lower). This fact suggests that the 5G network is honoring the priority parameter specified when these slices, with more severe QoS requirements, were enforced in the data plane. As a consequence, their traffic is treated with higher priority by the the Flow Agent, leading in a better delay performance and stability. To conclude the analysis of fig. 13, it can be noticed that the delay interval of the slices with less priority (use case 1 with mMTC traffic profile) is larger than the rest of the use cases. This interval measures the stability of the delay along the time (jitter). Results indicate that lowest priority traffic is buffered longer and forwarded with a more irregular pattern depending on the network congestion. It depends on how other slices with higher priority are consuming network resources. In the worst case scenario, the jitter varies roughly 250 μs (between 650 and 900 μs) for use case 1 which is a more than acceptable value for this kind of use cases. Concerning reliability, as with the experiments conducted with homogeneous traffic, the packet loss ratio is also 0% for all use cases so no graph is provided in this regard.

Fig. 14 depicts the system behaviour in terms of bandwidth when dealing with 5G IoT network traffic transmitted simultaneously from IoT devices of the 5 different use cases. The results correspond to a 10-second interval where the average bandwidth per slice has been computed at 0.25 second intervals. It can be observed that, for all use cases, the proposed framework is able to deliver a guaranteed bandwidth service within the boundaries agreed in the SLAs between the verticals and the service provider. Regarding the number of packets, the 5G infrastructure has managed around 1,2500,000 packets for this experiment while the average overall bandwidth obtained is 12.36 Gbps.

Therefore, it can be concluded that the proposed 5G IoT framework has been empirically validated against a large-scale infrastructure. It is suitable to deliver networking slicing for IoT services with guaranteed QoS requirements in 5G networks dealing with simultaneous and heterogeneous traffic. The framework has shown very promising results in terms of the number of slices supported, and in terms of the bandwidth, delay, jitter and packet loss able to be warrantied in the 5G infrastructure.

C. PERFORMANCE EVALUATION OF ADAPTIVE SLICE MANAGEMENT

Since 5G IoT networks are dynamic ecosystems subject to continuous changes, one of the most challenging features that a 5G IoT slice control framework must address is the flexibility to adapt to such complex scenarios. This implies a quick response in managing network slices and provisioning the appropriate network resources needed to enforce the slices in the data plane and meet the diverse requirements demanded by verticals. Fig. 15 provides a breakdown of the time spent by the proposed framework in creating different number of slices when ranging exponentially the number of slices from 2 to 256. The given results refer to the time consumed in the Resource Control plane (see Fig. 3), i.e., since the NBI of the SCA receives an intend-based message from the SDN controller until the slice has been successfully enforced in the data plane and an ACK message is sent back to the management plane. This process corresponds to steps 6 to 13 of the sequence diagram in Fig. 7 as discussed in subsection IV-E.

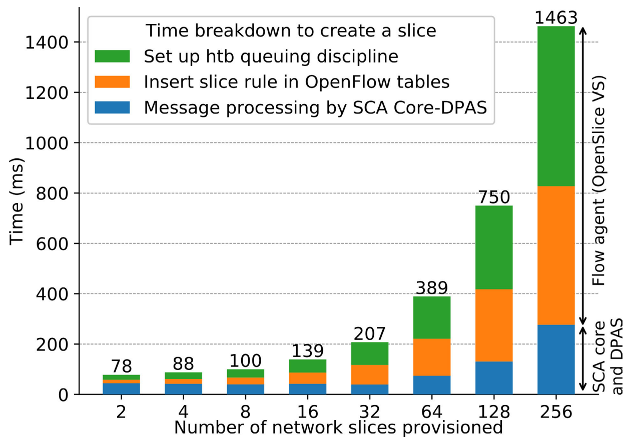


FIGURE 15. Empirical analysis of the scalability of the time required by the Resource Control plane to create a network slice (process the intend-based message and provision the necessary network resources).

Three architectural components are involved in this task: the SCA core, the DPAS module and the Flow Agent (OpenSliceVS). It can be observed that the major workload relies upon the Flow Agent. The Flow Agent is responsible for the insertion of the OpenFlow rule matching the slice traffic in the *slice definition and action table* (the orange stacked bar in Fig. 15) and setting up the htb queuing discipline where the slice traffic will be forwarded and the QoS requirements will be enforced (the green stacked bar in Fig. 15). Meanwhile, the time required by the SCA module to process the intend-based message has less impact on the overall time spent to create a slice (the blue stacked bar in Fig. 15).

The gathered results indicate that the proposed framework needs 1463 ms to create 256 slices, resulting in an average time of only 5.7 ms per slice. Similar experiments have been carried out to evaluate the time to delete and modify network slices, obtaining better results than those when creating new network slices (an average overall time of 21% lower). For the sake of brevity, these results are not shown. To delete or modify a network slice, the SCA core does not have to process a full intend-based message, instead it just delegates to the Flow Agent the required action based on the slice ID reported by the external policy enforcer of the management plane, which is faster. Similarly, regarding the Flow Agent side, releasing network resources or modifying previously provisioned network resources is quicker than provisioning them from scratch.

Hence, these results prove that the proposed 5G IoT framework is able to adapt on demand to evolving IoT traffic, adding, modifying or decommissioning in an efficient way (just a few milliseconds) the slicing policies imposed by the upper cognitive management layers.

VIII. CONCLUSION

This paper has presented a highly scalable and effective network slicing framework for 5G IoT networks, following the software-defined networking approach. All the specific requirements to provide network slicing capabilities in 5G IoT infrastructures identified in subsection III-A are fully addressed by our solution. It is capable of supporting key 5G features including network function virtualization, multi-tenancy, and device mobility. It is particularly designed for massive machine-type communications and is able to manage thousands of heterogeneous slices concurrently. Furthermore, it supports other 5G use cases that require enhanced mobile broadband, or ultra reliable and low-latency communications, or a combination. Five 5G IoT use cases covering these three ITU use case categories individually and collectively have been considered and their QoS requirements were abstracted to inform a realistic emulation in a testbed, where the proposed network slicing framework has been successfully tested and evaluated. Empirical results have highlighted the high scalability and performance achieved by this proposed system. In the tests, the implemented framework has been validated to be able to support mMTC traffic of up to 1 million IoT devices, achieve over 15 Gbps eMBB bandwidth, ensure delays in the order of μs for uRLLC, whilst providing high reliability in all of the tested scenarios with 0% packet loss ratio. Moreover, it takes only 5.7 ms on average to create a network slice in the data plane for challenging heterogeneous traffic situations.

Future work will investigate the application of the proposed solution in real-world 5G and beyond trials and evaluate the perceived quality of its performance for various vertical business applications. In addition, the integration of the OpenSliceVS Flow Agent with kernel bypass applications such as DPDK or eXpress will be assessed in order to mitigate the performance limitations of the Linux kernel network stack and, thus provide improved key performance indicators (KPIs) and boost scalability.

REFERENCES

[1] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2429–2453, 3rd Quart., 2018.

[2] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges," *Comput. Netw.*, vol. 167, Feb. 2020, Art. no. 106984. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128619304773

[3] L. U. Khan, I. Yaqoob, N. H. Tran, Z. Han, and C. S. Hong, "Network slicing: Recent advances, taxonomy, requirements, and open research challenges," *IEEE Access*, vol. 8, pp. 36009–36028, 2020.

[4] J. Cheng, W. Chen, F. Tao, and C.-L. Lin, "Industrial IoT in 5G environment towards smart manufacturing," *J. Ind. Inf. Integr.*, vol. 10, pp. 10–19, Jun. 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2452414X18300049

[5] W. Li, R. Liu, Y. Dai, D. Wang, H. Cai, J. Fan, and Y. Li, "Research on network slicing for smart grid," in *Proc. IEEE 10th Int. Conf. Electron. Inf. Emergency Commun. (ICEIEC)*, Jul. 2020, pp. 107–110.

[6] B. Dzogovic, B. Santos, J. Noll, V. T. Do, B. Feng, and T. V. Do, "Enabling smart home with 5G network slicing," in *Proc. IEEE 4th Int. Conf. Comput. Commun. Syst. (ICCCS)*, Feb. 2019, pp. 543–548.

[7] E. Kapassa, M. Touloupou, P. Stavrianos, and D. Kyriazis, "Dynamic 5G slices for IoT applications with diverse requirements," in *Proc. 5th Int. Conf. Internet Things: Syst., Manage. Secur.*, Oct. 2018, pp. 195–199.

[8] J. Ni, X. Lin, and X. S. Shen, "Efficient and secure service-oriented authentication supporting network slicing for 5G-enabled IoT," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 644–657, Mar. 2018.

[9] F. Kurtz, C. Bektas, N. Dorsch, and C. Wietfeld, "Network slicing for critical communications in shared 5G Infrastructures—An empirical evaluation," in *Proc. 4th IEEE Conf. Netw. Softw. Workshops (NetSoft)*, Jun. 2018, pp. 262–266.

[10] R. Trivisonno, M. Condoluci, X. An, and T. Mahmoodi, "MIoT slice for 5G systems: Design and performance evaluation," *Sensors*, vol. 18, no. 2, p. 635, Feb. 2018.

[11] C. Bektas, S. Bocker, F. Kurtz, and C. Wietfeld, "Reliable software-defined RAN network slicing for mission-critical 5G communication networks," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2019, pp. 1–6.

[12] S. Costanzo, I. Fajjari, N. Aitsaadi, and R. Langar, "Dynamic network slicing for 5G IoT and eMBB services: A new design with prototype and implementation results," in *Proc. 3rd Cloudification Internet Things (CIoT)*, Jul. 2018, pp. 1–7.

[13] Y. Tsukamoto, R. K. Saha, S. Nanba, and K. Nishimura, "Experimental evaluation of RAN slicing architecture with flexibly located functional components of base station according to diverse 5G services," *IEEE Access*, vol. 7, pp. 76470–76479, 2019.

[14] P. Salva-Garcia, J. M. Alcaraz-Calero, Q. Wang, J. B. Bernabe, and A. Skarmeta, "5G NB-IoT: Efficient network traffic filtering for multitenant IoT cellular networks," *Secur. Commun. Netw.*, vol. 2018, pp. 1–21, Dec. 2018.

[15] A. M. Escolar, J. M. A. Calero, and Q. Wang, "Highly-scalable software firewall supporting one million rules for 5G NB-IoT networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.

[16] Q. Wang, J. Alcaraz-Calero, R. Ricart-Sanchez, M. B. Weiss, and A. Gavra, "Enable advanced QoS-aware network slicing in 5G networks for slice-based media use cases," *IEEE Trans. Broadcast.*, vol. 65, no. 2, pp. 444–453, Jun. 2019.

[17] J. J. Diaz Rivera, T. A. Khan, A. Mehmood, and W.-C. Song, "Network slice selection function for data plane slicing in a mobile network," in *Proc. 20th Asia–Pacific Netw. Oper. Manage. Symp. (APNOMS)*, Sep. 2019, pp. 1–4.

[18] *System Architecture for the 5G System (Release 15)*, document TR 23.501 V15.0.03, Generation Partnership Project (3GPP), Dec. 2017.

[19] A. E. Kalor, R. Guillaume, J. J. Nielsen, A. Mueller, and P. Popovski, "Network slicing in industry 4.0 applications: Abstraction methods and end-to-end analysis," *IEEE Trans. Ind. Informat.*, vol. 14, no. 12, pp. 5419–5427, Dec. 2018.

[20] P. Rost, M. Breitbach, H. Roreger, B. Erman, C. Mannweiler, R. Miller, and I. Viering, "Customized industrial networks: Network slicing trial at hamburg seaport," *IEEE Wireless Commun.*, vol. 25, no. 5, pp. 48–55, Oct. 2018.

[21] T. Taleb, I. Afolabi, and M. Bagaa, "Orchestrating 5G network slices to support industrial Internet and to shape next-generation smart factories," *IEEE Netw.*, vol. 33, no. 4, pp. 146–154, Jul. 2019.

[22] 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," *Trans. Emerg. Telecommun. Technol.*, vol. 31, no. 1, Jan. 2020. Art. no. ett3726. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3726

[23] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "NFV and SDN—Key technology enablers for 5G networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2468–2478, Nov. 2017.

[24] C. Bouras, P. Ntarzanos, and A. Papazois, "Cost modeling for SDN/NFV based mobile 5G networks," in *Proc. 8th Int. Congr. Ultra Modern Telecommun. Control Syst. Workshops (ICUMT)*, Oct. 2016, pp. 56–61.

[25] A. Matencio-Escolar, Q. Wang, and J. M. Alcaraz Calero, "SliceNetVSwitch: Definition, design and implementation of 5G multi-tenant network slicing in software data paths," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 4, pp. 2212–2225, Dec. 2020.

[26] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, and A. Zhou, "The design and implementation of Open vSwitch," in *Proc. USENIX Symp. Netw. Syst. Design Implement.*, Oakland, CA, USA, May 2015, pp. 117–130. [Online]. Available: https://www.usenix.org/conference/nsdi15/technical-sessions/presentatio%n/pfaff

[27] The Linux Foundation Project. (2019). *Open vSwitch website*. [Online]. Available: https://www.openvswitch.org/

[28] October (2020). *Open Daylight Official Website*. [Online]. Available: https://www.opendaylight.org/
[29] The Open Infrastructure Foundation (OIF). (2020). *OpenStack Official Website*. [Online]. Available: https://www.openstack.org/
[30] *Study on Management and Orchestration of Network Slicing for Next Generation Network (Release 15)*, document TR28.801 V15.01.0, 3GPP, Mar. 2017.
[31] L. Cheng, "Network slicing architecture," Internet Engineering Task Force (IETF), Fremont, CA, USA, Tech. Rep. draft-geng-netslices-architecture-02, Jul. 2017.
[32] *Network Slicing for 5G Networks and Services (White Paper)*, 5G Americas, Bellevue, WA, USA, 2016.
[33] *Summary of Rel-15 Work Items (Release 15)*, document TR 21.915 V15.0.0, 3GPP, Sep. 2019.
[34] *Network Functions Virtualisation (NFV) Release 3;Evolution and Ecosystem; Report on Network Slicing Support with ETSI NFV Architecture Framework*, document GR NFV-EVE 012 V3.1.1, ETSI, Dec. 2017.
[35] *System architecture for the 5G System (5GS)*, European Telecommunications Standards Institute, Sophia Antipolis, France, Apr. 2019.
[36] ITU-R (Radio Communication Sector of ITU), *Minimum Requirements Related to Technical Performance for IMT-2020 Radio Interface(s)*, document ITU-R M.2410-0, ITU, Geneva, Switzerland, Nov. 2017. [Online]. Available: https://www.itu.int/dms_pub/itu-r/opb/rep/R-REP-M.2410-2017-PDF-E.pdf
[37] G. Architecture and W. Group. (2017). *View of 5G Architecture*. https://5g-ppp.eu/wp-content/uploads/2018/01/5G-PPP-5G-Architecture-Whi%te-Paper-Jan-2018-v2.0.pdf
[38] M. Ayaz, M. Ammad-Uddin, Z. Sharif, A. Mansour, and E. M. Aggoune, "Internet-of-Things (IoT)-based smart agriculture: Toward making the fields talk," *IEEE Access*, vol. 7, pp. 129551–129583, 2019.
[39] K. Tange, M. De Donno, X. Fafoutis, and N. Dragoni, "A systematic survey of industrial Internet of Things security: Requirements and fog computing opportunities," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 4, pp. 2489–2520, 2nd Quart., 2020.
[40] T. Spadini, D. L. de Oliveira Silva, and R. Suyama, "Sound Event Recognition in a Smart City Surveillance Context," 2019.
[41] IBM Watson Media Support Center. (Oct. 2020). *Internet Connection and Recommended Encoding Settings*. [Online]. Available: https://support.video.ibm.com/hc/en-us/articles/207852117-Internet-conn%ection-and-recommended-encoding-settings/
[42] J. Egger and T. Masood, "Augmented reality in support of intelligent manufacturing – a systematic literature review," *Comput. Ind. Eng.*, vol. 140, Feb. 2020, Art. no. 106195. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0360835219306643

ANTONIO MATENCIO ESCOLAR is currently pursuing the Ph.D. degree with the University of the West of Scotland, U.K. He is currently a member of the B5G-Hub Research Group, University of the West of Scotland. He has been actively involved in the H2020 5G-PPP Slicenet project. His main research interests include network slicing, software datapath, IoT, SDN, data plane programmability, and 5G mobile networks and network control and management.

JOSE M. ALCARAZ-CALERO (Senior Member, IEEE) received the Ph.D. degree in computer Science from the University of Murcia. He is currently a Full Professor in next-generation networks and security with the University of the West of Scotland. He is the Technical Co-Coordinator of the EU H2020 5G-PPP SELFNET and SliceNet projects and a Co-Principal Investigator of the EU H2020 5G INDUCE and 6G BRAINS projects. His professional interests include network cognition, management, security and control, service deployment, automation and orchestration, and 5G mobile networks.

PABLO SALVA-GARCIA received the Ph.D. degree from the University of the West of Scotland, U.K. He is currently a Post-Doctoral Researcher with the University of the West of Scotland. He has been technically involved in the EU Horizon H2020 SELFNET and SliceNet projects. He is a member of the B5G-Hub researching group. His main interests include network management, cognitive control plane, data plane programmability, software-defined networks, and video delivery in 5G networks.

JORGE BERNAL BERNABE received the B.S., M.S., and Ph.D. degrees in computer science and the M.B.A. degree from the University of Murcia, Spain. He is currently an Assistant Professor with the University of Murcia. He has been a Visiting Researcher with the Hewlett-Packard Laboratories and the University of the West of Scotland. During the last years, he has been working in several European research projects, such as SocIoTal, ARIES, OLYMPUS, ANASTACIA, INSPIRE-5G, and CyberSec4EU. He has authored several book chapters and more than 60 articles in international top-level conferences and journals.

QI WANG received the Ph.D. degree in mobile networking from the University of Plymouth, U.K. He is currently a Full Professor with the University of the West of Scotland. He is the Technical Co-Coordinator of the EU H2020 5G-PPP SELFNET and SliceNet projects and a Co-Principal Investigator of the EU H2020 5G INDUCE and 6G BRAINS projects. He is a Board Member of the Technology Board of EU 5G-PPP. His research primarily focuses on 5G mobile networks, video networking, and artificial intelligence.