LTE as a Service: leveraging NFV for realising dynamic 5G network slicing

Luis Sanabria-Russo†, Ludovico Righi*, David Pubill†, Jordi Serra†, Fabrizio Granelli*, Christos Verikoukis†
†Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA)
*Department of Information Engineering and Computer Science, University of Trento
{lsanabria, dpubill, jserra, cveri}@cttc.es
ludovico.righi@studenti.unitn.it, fabrizio.granelli@unitn.it

***Abstract*—Advances in virtualisation technology have reached the mobile networking domain. Network Functions Virtualisation Management and Orchestration (NFV MANO) as proposed by ETSI and realised via Opensource MANO (OSM) allows sharing or partitioning a fairly generic pool of hardware into virtual compute, network and storage resources among differentiated services. A direct consequence of this is a dramatic reduction in CAPEX/OPEX, but also the possibility of instantaneously deploy network services across one or several Mobile Network Operators' (MNO) infrastructure. This work demonstrates how the upcoming fifth generation (5G) of mobile communications envisions NFV MANO for instantiating network services, including Core Network components, and wireless SDN controllers for enforcing end-to-end QoS policies via network slices that span from the radio access segment to the backend packet network.**

***Index Terms*—NFV, OpenAirInterface, MANO, FlexRAN, 5G**

I. Introduction

5G promises to change how society consumes and shares information by providing tools to achieve the *Internet of Everything* vision, where the number of devices attaching to a single base station is likely to increase from tens of thousands to several hundred of thousands [1]. Furthermore, projections based on 2G to 4G trends estimate 5G's key performance indicators (KPI) to reach 10 Gbps downlink (DL) data rates, and 1 ms latencies for ultra low latency use cases [2], [3].

Realising such KPIs calls out for important design changes on the communication infrastructure. First, the virtualised 5G core network components need to be pushed closer to the network edge and allow for fast instantiation in order to provision the radio access segment of the network (RAN). And second, the ability to provide service level agreements (SLA) via virtual end-to-end partitions (slices) would allow 5G to realise vertical industries use cases, such as Industry 4.0 applications [4].

Software Defined Networks (SDN) and Network Functions Virtualisation (NFV) are thought to be key enablers for the aforementioned vision. Indeed, among several advantages, they allow operators to create several differentiated services as virtual overlays on top of a generic pool of hardware resources. Jointly, SDN and NFV are expected to yield important CAPEX and OPEX reductions, but also enable dynamic reconfiguration of services and a centralised control plane.

This work leverages OpenAirInterface (OAI) and Core Network (OAI-CN, or CN) [5], [6], as well as ETSI's NFV Management and Orchestration (MANO) framework [7], [8] to deploy and configure 5G end-to-end network slices at instantiation time. Furthermore, it explores the role wireless SDN Controllers (i.e. FlexRAN [9]) play as SLA enforcers by developing a priority traffic detector which triggers the migration of a specific user to a priority slice at runtime.

Section II provides substantial background related to NFV, as well as briefly describes contributions similar to this one. The proposed system architecture is contained in Section III. Use case definitions and implementation, as well as analysis of the results are discussed in Section IV. Conclusions and future directions are presented at the final Section V.

II. Background

The 5G vision proposes a flexible network design, such that a single physical infrastructure is shared among different applications, while service requirements are guaranteed leveraging technologies such as SDN and NFV.

NFV and SDN are complementary technologies that achieve the level of abstraction and flexibility required to satisfy stringent applications' requirements while maximising network infrastructure reutilisation. Specifically, NFV decouples physical network functions (PNFs) (e.g.: firewalls, routers, load-balancers, etc.), from dedicated hardware by implementing the same functionality in software, coined virtual network functions (VNFs). VNFs may then be instantiated in data centres at backend clouds, or on top of devices equipped with compute and storage resources at the edge [10]. Its specifications could be modified according to requirements or load, and then decommissioned when no longer needed; freeing compute, network and storage resources for other VNFs.

SDN eases network management through a softwarisation approach. Namely, it decouples the data plane from the control plane, centralising network management in a so-called SDN controller. With a global view of the network resources, SDN controller applications (SCA) can take advantage of the numerous southbound interfaces (e.g.: OpenFlow [11], NET-

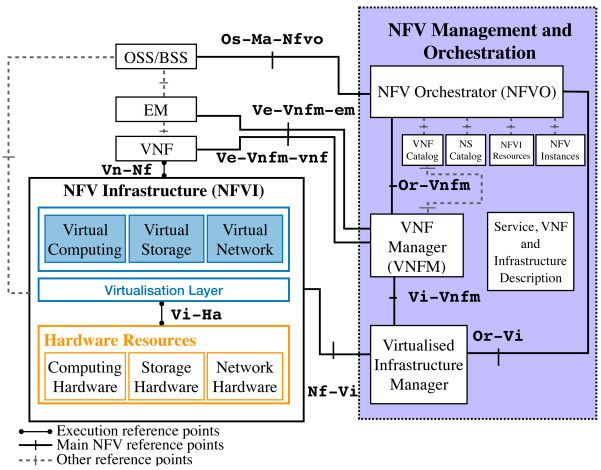

Fig. 1. ETSI NFV reference architectural framework [7]

CONF [12], among others) to gather network state information and act upon each forwarding device (i.e. PNF or VNF) configuration accordingly, e.g. by establishing data flow paths that guarantee certain QoS requirements for a specific service. Together, SDN and NFV enable dynamic compute and network resources allocation for heterogeneous QoS requirements, which helps to circumvent the undesired effects of a changing network environment on sensible 5G applications, such as haptic communications, industry 4.0, autonomous driving, robotics, manufacturing, among others.

ETSI's efforts towards the standardisation of the 5G vision has yielded the Network Functions Virtualisation Architecture (NFVA) [7], which leverages the dynamism, flexibility and reusability provided by SDN and NFV primitives. NFVA's several components handle the lifecycle and interconnection of VNFs in order to expose virtual Network Services (NS) to applications. ETSI's NVFA is shown in Figure 1, including its main components: NFV Infrastructure (NFVI), Virtual Infrastructure Manager (VIM), VNF Manager, and the NFV Orchestrator (NFVO).

A. Virtual Infrastructure Manager (VIM)

Inside the NFVA, the VIM is responsible for the control and management of the interaction between VNFs and the NFVI hardware resources, such as compute, storage and network, as well as their virtualisation. It takes care of exposing a pool of virtualised resources derived from the NFVI, as well as allocating such resources to VNFs. VIMs also manage virtual network overlays to connect VNFs using SDN. In this work, OpenStack Stein[1] is used as VIM.

B. VNF Manager

It is responsible for VNF lifecycle management. That is, the instantiation, scaling, and termination of one or several VNFs. State of the art VIMs often include a service for VNF Management, like Tacker in OpenStack[2].

[1]https://docs.openstack.org/stein/

[2]OpenStack Tacker: https://wiki.openstack.org/wiki/Tacker

C. NFV Orchestrator (NFVO)

The realisation of a NS is carried out by the NFVO, which is able to gather information about the NFVI from one or several VIMs. Moreover, information regarding the available VNFs (through a collection of descriptors on-boarded by the corresponding NFVI administrators), allocated resources, performance metrics about VNFs and virtual links, NFVI faults (outage) information, among others [8] could be used to monitor and update NS.

NFVO deploys NS by embedding what is defined in VNF and NS descriptors (VNFd, and NSd, respectively) into the NFVI. That is, the allocation of the required virtual networking, compute and storage resources to start VNFs and allow communication between them. This implies the placement of the VNFs into virtualisation containers, e.g., Virtual Machines (VM), at the most convenient compute nodes[3] within the NFVI. This work leverages ETSI's own Open-source MANO (OSM) release FIVE[4] as NFVO.

D. Related work

Several related works have dealt with the problem of network slicing in mobile networks from a practical implementation viewpoint [13], [14]. These works make use of OAI [5], [6] to implement the building blocks of the RAN and Core Network (CN) in a testbed. All of them considered a Cloud RAN (C-RAN) architecture for the RAN, where the functionalities of the eNodeB are split between a Radio Cloud Centre (RCC) and a Radio Remote Unit (RRU). The RCC implements most of the eNodeB functionalities, namely, PHY, MAC, Radio Link Control (RLC); while I/Q samples are transmitted/received via an Ethernet link to/from the RRU, which implements the RF functionalities. Moreover, the RCC interacts with an SDN controller based on FlexRAN [9] that allows for dynamic network slicing. Namely, it configures and dynamically assigns radio resources to slices according to the network state and/or QoS requirements.

Herein, as in [13], [14], OAI and FlexRAN software are leveraged to implement the building blocks of a mobile network, and to manage the radio resources within a network slicing framework, respectively. However, unlike [13], [14] this work makes use of NFV MANO to virtualise the building blocks of the CN, consequently resulting in a more flexible and programmable approach. This work's contributions are summarised in the following:

- Deploys end-to-end 5G network slicing capabilities leveraging open source software following ETSI's NFV MANO framework.
- Implements scenarios where user priority is required. Consequently solving such requirements via network slicing.
- Evaluates VNF instantiation delay for CN components.

[3]The one satisfying the requirements stated on the corresponding descriptors.

[4]https://osm.etsi.org/

- Proposes a method for achieving dynamic inter-slice user migrations based on traffic detection.

The following Section III will detail ETSI's NFV MANO framework, which is used for instantiating all OAI-CN components, backend network slices and a wireless SDN Controller for configuring and managing RAN slices.

III. LTE as a Service

This section describes the architecture of the platform used in this work, dubbed LTE as a Service (LTEaaS).

A. System architecture

The virtualisation of Core Network (CN) components is one of the key enablers for orchestration in LTEaaS. Leveraging OAI-CN, each element (e.g.: MME, HSS, SPGW) can be realised as a single VM that uses virtual links to exchange control information with other CN components. The same is true for wireless SDN controllers, such as FlexRAN. Figure 2 describes the architectural framework upon which the evaluation will be performed.

On the right of Figure 2 it is possible to devise the NFV MANO framework, alongside its respective NFV reference points for message exchange (e.g.: network service orchestration, NFVI/VNF metrics gathering, etc.). As the base for all virtualisation, the NFVI (bottom centre of the figure) abstracts a pool hardware into virtual compute, storage and network resources for VNFs. All OAI-CN components and services are located on different backend slices, mimicking a multi-tenancy scenario. The communication between the non-virtualised RAN (left side of the figure) and the CN is done through a Physical Network Function (PNF), i.e. a switch; the NFVI and NFV MANO run on top of 3 24-CPUs mainframes with Ubuntu Server 16.04.6; while the National Instruments USRP B210 is used as a 4G RRU controlled by OAI.

B. RAN and Wireless SDN Controller

A typical OAI deployment delegates eNodeB functionality to a general purpose computer with a high-speed USB-3 connection to a software-defined radio, e.g. USRP. The RAN section in Figure 2 shows a RAN controller (OAI Node), a RRU implemented on an USRP B210, and a set of users as UEs[5]. The OAI-Node is connected through a PNF to the same VLAN as the Service OAI-CN slice, which allows packet network access to UEs via a GPRS Tunnelling Protocol (GTP) tunnel [15] ending at OAI-CN's SPGW.

RAN slices creation, termination, modification and management is performed by the wireless SDN Controller (FlexRAN) at the OAI-CN. FlexRAN exposes a RESTful northbound interface (NBI) that allows developers quick access to radio level metrics, as well as other eNodeB or UE metrics, such as per UE Packet Data Convergence Protocol (PDCP) or MAC statistics. Relevant slice configuration parameters supported by FlexRAN also include DL/UP bandwidth limits in terms of resource blocks (RB), and live slice modification and migration of specific UE(s) to a determined slice [16].

[5]UEs are small Raspberry Pi 3 model B with a Huawei LTE E3372 USB dongle.

C. Orchestration

The service provided by LTEaaS is an instance of the "Virtualised OAI-CN and backend slices" block of Figure 2. That is, VNFd and NSd describing such network services must be developed and on-boarded onto the NFVO. Moreover, such descriptors should take into account the software dependencies, network addressing, hostnames, and other specific OAI-CN configuration files. Furthermore, services in backend slices should also be considered in the NSd, alongside the corresponding configuration for each APP.

In order to prepare each VNF to realise the required service, Day-0 configuration techniques for orchestration were performed, i.e. cloud-init[6]. Such type of configuration is executed the first time a VNF boots up. It allows developers to customise generic cloud images according to application requirements at boot time and without the need for rebooting the system[7]. For LTEaaS, each VNFd includes a cloud-init file that downloads dependencies and the required software right after instantiation.

Picking a solely Day-0 configuration approach for LTEaaS may suppose an increased instantiation time when compared with deploying pre-configured images for each VM. Nevertheless, the flexibility offered by such approach (cloud-init) allows developers to keep up-to-date with system/application upgrades, makes reconfiguration easier for researchers, and saves considerable storage space at the VIM[8].

IV. Evaluation

To realise the benefits of the proposed infrastructure two main tests are proposed, namely instantiation delay tests, and user/slice management tests.

A. Instantiation delay

The key characteristic of the architecture used in this work is its suitability for NFV orchestration. That is, an Admin element (see Figure 2), e.g. a script or a manual RESTful client, should able to reach NFV MANO's NBI and trigger the instantiation of the Virtualised OAI-CN and backend slices. In this work, two orchestration approaches are followed: 1) Day-0 configuration only, and 2) Pre-configured images. The former relies on cloud-init and complementary scripts for customising default Linux cloud images[9] into OAI-CN components (and Services' APPs); while the latter uses images that were configured beforehand.

[6]Cloud-Init: The defacto multi-distribution package that handles early initialisation of a cloud instance https://cloudinit.readthedocs.io

[7]Configurations such as changing the Linux kernel require a reboot. This is the case of the SGPW component of the OAI-CN.

[8]Mostly because a heavily customised VM could be conceived from a fairly light and generic cloud image.

[9]Ubuntu 18.04 cloud images, available at: https://cloud-images.ubuntu.com/.

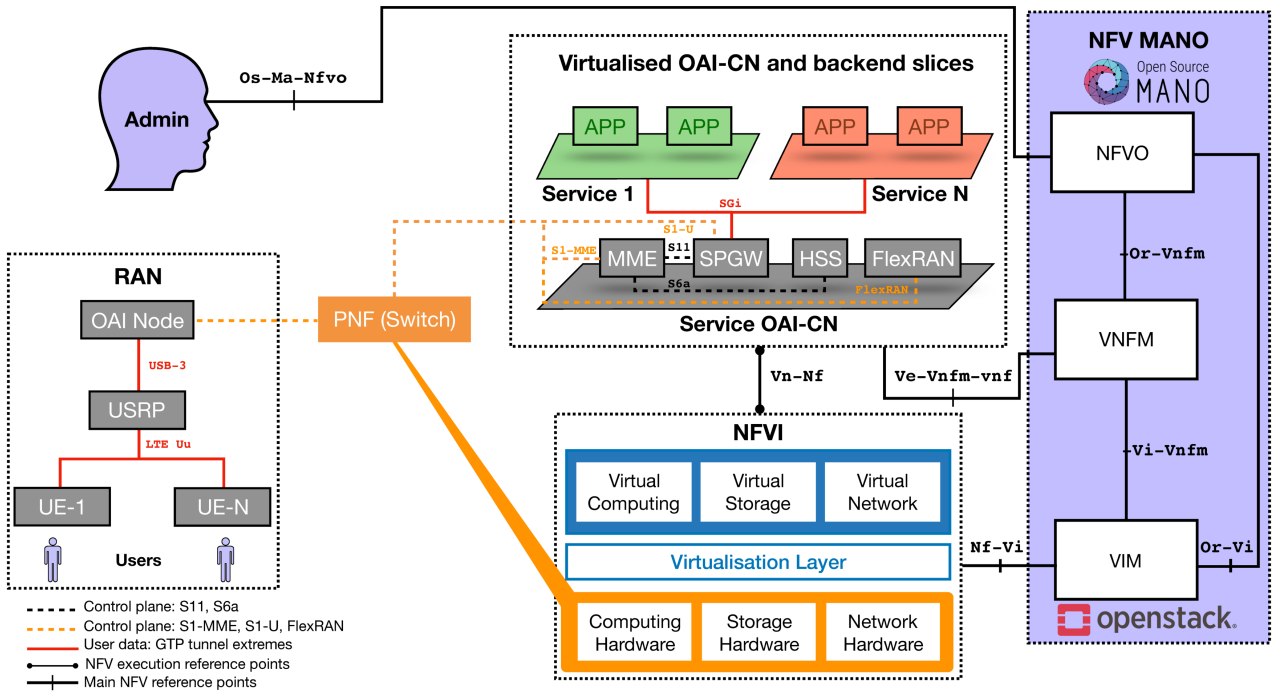

Fig. 2. LTEaaS reference architecture

TABLE I
INSTANTIATION DELAY (Δt): AVERAGE RESULTS FROM 20 ORCHESTRATIONS

Configuration method\Statistics	$\mu_{\Delta t}$ (s)	$\sigma_{\Delta t}$ (s)
Day-0 configuration only	270.61	33,66
Pre-configured images	63.28	2,41

The instantiation delay, Δt, refers to the total elapsed time between the Admin triggering NFVO's NBI, and the successful instantiation of OAI-CN and all of its components. Table I collects the average results from 20 instantiations for each of the methods described. It is clear from the results that having pre-configured cloud images is the fastest way of instantiating virtual network functions. On the other hand, relying only on Day-0 configuration increases the instantiation time for several reasons, e.g. varying Internet connection speed, the need for downloading dependencies, reboots, among others; nevertheless, it allows for easier customisation, making it ideal for testing configuration changes or software versions.

In the end, Table I suggests that if fast orchestration of OAI-CN components were the main goal, it is better to have pre-configured images instead of relying on pure Day-0 configuration (around 4.5x faster).

B. *Users and Slices' management*

This section describes a set of scenarios where RAN slices' resources are managed in order to provide more bandwidth to (predefined) priority users. Three scenarios are defined:

1) Slice RB reduction: redistribution of RB associated to a determined slice may free up resources to other, higher priority slices. In this reference scenario, two UEs are initially placed on different RAN slices that evenly share the amount of RB for downlink (DL) traffic. Two backend network slices are provisioned with iPerf [17] servers that work as source for the downlink traffic received at each UE. Figure 3, shows a simplified view of the topology of this scenario based on Figure 2.

Backend iPerf servers are set to transmit downlink UDP streams at 8 Mbps, while the DL RB for both RAN slices are evenly distributed. Then, an instruction to switch from 50%/50% (even) to 75%/25% is triggered by the system administrator (through the Admin element in Figure 3), which is reflected on the reduced maximum download throughput achieved by the node in the affected slice (UE-1). The moment of the switch is made evident by Figure 4.

2) User migration to priority slice: RAN slices may also be configured to provide different bandwidth configurations for privileged users, or to enforce QoS. In this scenario, two RAN slices are configured:

- Slice 1: only 10% of DL RB.

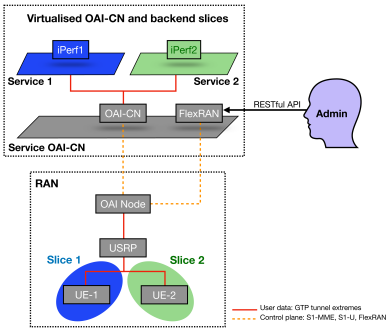

Fig. 3. Slice RB reduction scenario

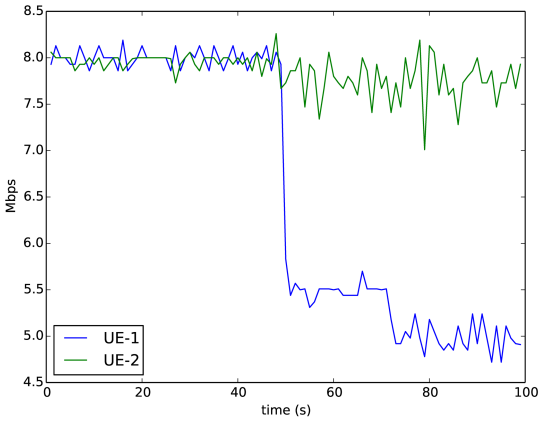

Fig. 4. Slice RB reduction scenario: Cutting UE-1's RAN slice DL RB in half

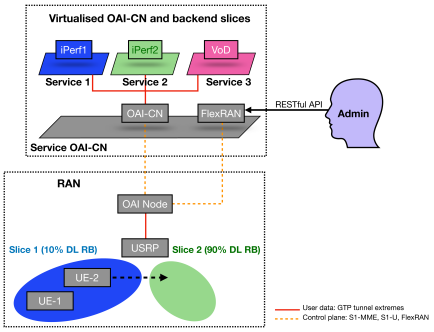

Fig. 5. User migration to priority slice scenario: UE-2 is manually moved to Slice 2 when video playback starts

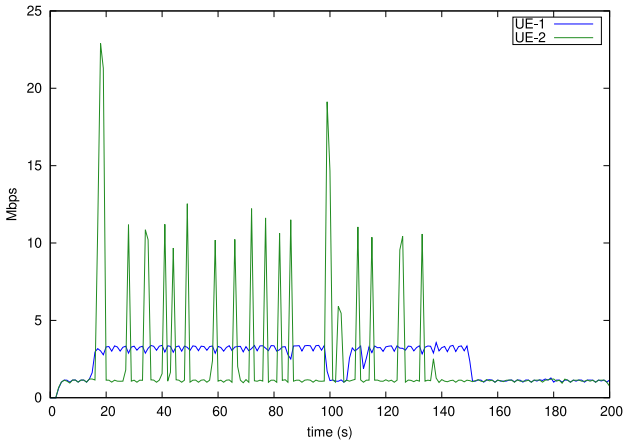

Fig. 6. User migration to priority slice scenario: UE-2 is manually moved to Slice 2 when video playback starts at around 10 s

- Slice 2: the other 90% of DL RB.

The above implicitly means that Slice 2 is the one with better downlink channel bandwidth.

In this scenario, a video on demand (VoD) server is instantiated on an additional backend slice, as shown in Figure 5. Both users at the RAN start downlink UDP flows from a backend iPerf server at 1 Mbps, mimicking background traffic. Later, after 10 seconds both UEs simultaneously start to play a video from the VoD server. Consequently, FlexRAN's NBI is triggered via the Admin element to switch UE-2 from Slice 1 to Slice 2 (higher %DL RB), which makes video buffering go faster than UE-1's.

Figure 6 shows the downlink throughput for each UE. Before the 10 s mark, it is possible to see the downlink throughput at 1 Mbps for both UEs, which represents the iPerf traffic. When the instruction to move UE-2 to Slice 2 is triggered (around 10 s), video buffering is shown as spurs of downlink traffic for this user. On the other hand, UE-1 is forced to buffer the video at lower speeds, what results in a prolonged (albeit at lower rate) buffering period. Around the 150 s mark buffering ceases, so both users' DL throughput line returns to the iPerf "floor" of 1 Mbps.

3) User migration to priority slice based on automatic detection of DL throughput increase: dynamic resource allocation or inter-slice user migration is a promising feature for many 5G verticals. In the presence of priority users, a dynamic slicing framework should be able to detect the resources demands and adjust the RAN accordingly. This scenario's topology is essentially the same as the one shown in Figure 5, with the exception that the user migration is automatically executed by a basic algorithm (see Algorithm 1) that detects any increase in the downlink throughput from a particular UE via FlexRAN's NBI, and then switches that specific user to a priority slice.

UEs are setup to download generic UDP streams from an iPerf server at 1 Mbps for at least 30 s. Then, an arbitrary user (UE-2) starts a video playback from the backend VoD server. Figure 7 shows UE-2's downlink throughput spurs corresponding to the video buffering in Slice 2 (similar to Figure 6), with no intervention for the system administrator.

V. Conclusions and Future directions

The agility provided by NFV MANO represents the key for provisioning core network components, network slices, applications, and other virtual network functions (VNF) such as wireless SDN Controllers. Indeed, joint usage of SDN and NFV is expected to help provide the required infrastructure for supporting the surge of connected devices and stringent KPIs expected in 5G.

This work demonstrated such functionality, putting specific emphasis on instantiation delay and

Algorithm 1: Inter-slice user migration based on downlink (DL) throughput increase

```
new F; // FlexRAN NBI
new Rx_queue; // Hash w/ circular array size 30
new μ, σ; // mean and std, respectively
while True do
    for ue in F.getUes() do
        DL_throughput = F.getDLTh(ue);
        if Rx_queue[ue].isFull() then
            μ =Rx_queue[ue].getMean();
            σ =Rx_queue[ue].getStd();
            if DL_throughput ≥ μ + 3σ then
                F.switchSlice(ue);
                Rx_queue[ue].clear()
        else
            Rx_queue[ue].append(DL_throughput);
    sleep(1);
```

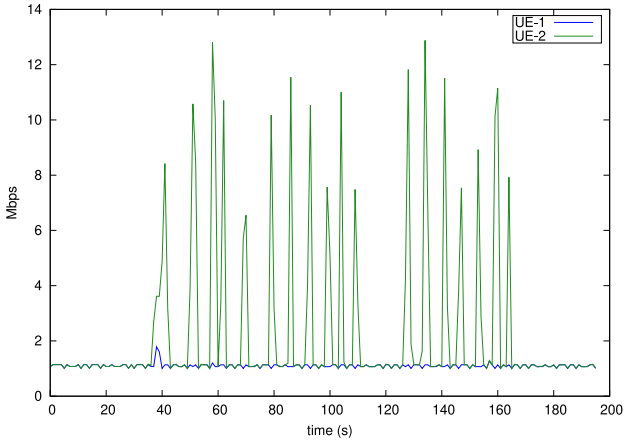

Fig. 7. User migration to priority slice based on automatic detection of traffic

user/RAN slices management. Results highlight the flexibility offered by Day-0 configuration for VNFs and the reduced instantiation delay provided by pre-configured images. Furthermore, it explores RAN slices management and configuration through the use of a wireless SDN Controller, i.e. FlexRAN, and how developers may leverage FlexRAN's NBI to automate different RAN slices-related tasks, e.g.: management of UL/DL RBs, inter-slices user migrations, and RAN-related metrics gathering.

Based on the above considerations, the proposed work can be considered as the base implementation for future updates related to 5G Core Network, and more relevantly, radio access functional splits. The former generally implies new CN components that comply with 3GPP Release 15 and beyond, while the latter allows for the decoupling of RAN-related tasks, e.g.: Radio Resource Control (RRC), Packet Data Convergence Protocol (PDCP), Radio Link Control (RLC), and MAC. By using the proposed architecture, it is possible to provide independent placement and scaling of such components to face dynamic environments or to comply with different policies, thus paving the way for the advent of the 5G Service Based Architecture and network slicing.

Acknowledgements

This work has been funded by the following research projects: 5G-Solutions (856691).

References

[1] I. Quintana-Ramirez, A. Tsiopoulos, M. A. Lema, F. Sardis, L. Sequeira, J. Arias, A. Raman, A. Azam, and M. Dohler, "The making of 5g: Building an end-to-end 5g-enabled system," *IEEE Communications Standards Magazine*, vol. 2, no. 4, pp. 88–96, December 2018.

[2] M. Dohler, T. Mahmoodi, M. A. Lema, and M. Condoluci, "Future of mobile," in *2017 European Conference on Networks and Communications (EuCNC)*, June 2017, pp. 1–5.

[3] A. Aijaz, M. Dohler, A. H. Aghvami, V. Friderikos, and M. Frodigh, "Realizing the tactile internet: Haptic communications over next generation 5g cellular networks," *IEEE Wireless Communications*, vol. 24, no. 2, pp. 82–89, April 2017.

[4] Huawei, et al, "5G Service-guaranteed network slicing white paper," https://www-file.huawei.com/-/media/corporate/pdf/white%20paper/5g-service-guaranteed-network-slicing-whitepaper.pdf?la=en.

[5] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet, "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 33–38, 2014.

[6] OpenAirInterface, "5G software alliance for democratising wireless innovation," https://www.openairinterface.org/.

[7] ETSI NFV ISG, "Network Functions Virtualisation (NFV); architectural framework," https://www.etsi.org/deliver/etsi_gs/nfv/001_099/002/01.02.01_60/gs_nfv002v010201p.pdf.

[8] ——, "Network Functions Virtualisation (NFV); Management and Orchestration," https://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf.

[9] X. Foukas, N. Nikaein, M. M. Kassem, M. K. Marina, and K. Kontovasilis, "FlexRAN: A flexible and programmable platform for software-defined radio access networks," in *Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies*. ACM, 2016, pp. 427–441.

[10] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[11] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[12] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman, "Rfc 6241, network configuration protocol (netconf)," *IETF*, 2011.

[13] S. Costanzo, I. Fajjari, N. Aitsaadi, and R. Langar, "A network slicing prototype for a flexible cloud radio access network," in *IEEE annual consumer comm. and networking conference*. IEEE, January 2018.

[14] ——, "Dynamic network slicing for 5G IoT and eMBB services: a new paradigm with prototype and implementation results," in *Cloudification of the Internet of Things (CIoT)*. IEEE, July 2018.

[15] ETSI TS, "Universal Mobile Telecommunications System (UMTS); LTE; General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U) (3GPP TS 29.281 version 15.3.0 Release 15)."

[16] Mosaic5G, "FlexRAN northbound API documentation," http://mosaic-5g.io/apidocs/flexran/.

[17] J. Dugan et al, "iPerf - The ultimate speed test tool for TCP, UDP and SCTP," https://iperf.fr/.