

End-to-end Network Slicing in Support of Latency-sensitive 5G Services

Rafael Montero¹[0000-0002-5338-2941], Fernando Agraz¹[0000-0001-8958-0088], Albert Pagès¹[0000-0002-6661-3630], Salvatore Spadaro¹[0000-0002-4100-1726]

¹ Universitat Politècnica de Catalunya (UPC), Jordi Girona 1-3, 08034, Barcelona, Spain
rafael.montero@tsc.upc.edu

Abstract. Network slicing has been taking a major role in upcoming 5G network implementations. However, in order to provision and maintain end-to-end slices, the management and orchestration among different network segments is required. As a result, techniques and components have risen to fulfil these tasks. In this work, we present latency-aware slicing, which is enabled by the provisioning of network slices equipped with an end-to-end latency sensor. This sensor is added to the service chain, allowing for real time monitoring and eventually actuation upon latency requirements violations. Moreover, we introduce an architecture capable of handling the deployment of such sensors while also coordinating the provisioning of the slice across optically interconnected DCs. To experimentally demonstrate the deployment of a slice with latency sensing we set up a multi-segment testbed connecting client VMs. The presented results demonstrate the behavior of the latency sensor and how it enables latency optimization through path reconfiguration.

Keywords: 5G, Network Slicing, Slice Composition, Latency Sensor, Service Chaining, Optical Networks.

1 INTRODUCTION

The current trend towards enabling legacy network scenarios up to the 5G standards has become a challenge to network operators. Especially in the case where highly demanding service types such as enhanced mobile broadband (eMBB), ultra-reliable and low latency communications (URLLC), and massive machine-type communications (mMTC) arise with different requirements. Despite of this, the operators need to deploy these kinds of services over the same physical infrastructure, thus sharing the same network resources, but with the responsibility of maintaining a level of isolation between them, as well as guaranteeing a proper functionality according to their particular requirements.

In this regard, Network Slicing has come to help accommodate networks to this behavior. This technology entails that a physical network can be partitioned (i.e., sliced), to enable different services with different requirements to coexist using the same underlying network resources. A slice in turn, has to be able to allocate multiple services for a particular tenant. This way, a tenant is then able to deploy ser-

vices/slices over an infrastructure that is shared with other tenants. To do this, a higher layer entity is required. Such entity takes charge of the slice management, as well as communicating the required configurations to each segment control entity to guarantee slice isolation. It is worth noting here that slice isolation can be either physical or virtual.

Another important aspect in 5G is the use of a virtualized approach for network function deployment, considering the introduction of Virtual Network Functions (VNFs). Such virtual functions are intended to provide specific functionalities according to service requirements. In this way, a service could be composed of multiple VNFs, which may be allocated in different physical segments of the infrastructure. Hence, some level of interconnection is required. The reachability between VNFs leads to the Service Chaining concept, which entails functions to be interconnected (i.e., chained) in a specific order to accomplish overall network service functionality.

Deploying a service composed by VNFs brings many other challenges. As noted before, such functions can be allocated in different network segments according to resource availability or functional requirements. Again, a high layer entity controlling slice provisioning is required to provide coordination between different segments and/or different technologies. To accomplish this, a new component (namely the NFV coordinator) is presented in this work to enable the communication between different network segments and components to provide multi-segment slices and their associated service chain.

Besides the deployment of a slice, it is also important to consider its maintenance over time according to specific client requirements. In this regard, latency appears as one of the most demanding requirements in the services defined in the 5G ecosystem. As a matter of fact, previous works have already considered it for slice provisioning [1]. However, real-time monitoring data becomes mandatory for any slice that has latency constraints. Gathering such monitoring data and further analyzing it, paves the way for management systems to react whenever latency levels reach non-desired points. The next step is to actuate according to the analyzed data. In this matter, it is possible to consider either acting reactively, preventively or predictively. In this work, we use a policy-based approach to perform preventive actions over the network (acting before service degradation), by putting emphasis on monitoring latency levels.

In light of the above, in this work we propose the use of a sensor (i.e., monitoring entity) in the form of a VNF to gather latency information from a running slice instance. The remainder of the paper is structured as follows, in Section II we present the architecture used, which evolves our previous work in [2]. Then, in Section III, an in-depth analysis of the used latency sensing mechanism is given. Section IV presents the experimental testbed that has been used for the functional validation, while Section V finally rounds up the main achievements of this work.

2 PROVISIONING AND MAINTENANCE OF NETWORK SLICES IN A MULTI-SEGMENT ENVIRONMENT

As introduced in the previous section, a Network Slice (NS) may demand its provisioning over different segments of the underlying infrastructure. This entails that high level coordination of network components is required in order to fulfil slice deployment. Furthermore, slice maintenance is also a high priority, as it should be conformant with the demanding requirements established for 5G services. Therefore, both stages of the slice lifecycle must be handled by an architecture capable of providing such guarantees and functionality. In this paper we present an architecture for the deployment and maintenance of slices in a multi-segment scenario, considering as well a set of techniques and tools to achieve the discussed necessities.

The basics of the proposed architecture along with details regarding the behavior of each of its components are presented in the following subsection. Moreover, the newly implemented modules at the Management and Orchestration (MANO) and Software Defined Networking (SDN) control level are also introduced.

2.1 Architecture

Fig. 1 depicts the overall architecture utilized in this work to enable multi-segment provisioning and maintenance of network services. As shown in the figure, the focus is on enabling optically interconnected Datacenter (DC) segments, in this case Multi-access Edge Computing (MEC) DC and Core DC, to allocate the required slice computational resources while considering as well their interconnection across the existent optical network segments (i.e., Metro/Access and Core). In terms of computational resources, a slice can be composed of several VNFs with different functionalities, which may also be chained in a specific order in spite of their physical location.

Accomplishing such performance requires involvement of different software components. At the lowest level, each network segment is exposed and configured through its own network control and management components (i.e., SDN Controller and/or Network Orchestrator). In the case of DC segments, which consider both computational and network resources, the controller allows configuring the DC network while the orchestrator is in charge of managing VNFs/Virtual Machines (VMs). On the other hand, optical network segments merely require network resources to be configured and exposed. In such case, only a SDN controller may be required. From a higher level point of view, the whole management of a compute plus network composed segment (e.g., DC) is seen as Virtual Infrastructure Management (VIM), while in the case of the optical network segments can be referred as Wide Area Network (WAN) Infrastructure Management (WIM).

Above the network segments, a MANO entity provides virtual network infrastructure as proposed by the ETSI in [3]. The Network Function Virtualization Orchestrator (NFVO) provides a set of blueprints for the NS and VNF creation. Hence, upon the request from an external client, the NFVO becomes in charge of orchestrating the whole lifecycle of the slice. The VIM Manager component in turn, allows registering each VIM so they become available for network slice deployment. Moreover, it is in

charge of coordinating requests to each VIM coming from the NFVO. Additionally, the VNF Manager enables configuring VNFs at both slice provisioning and runtime stages.

To enable coordination between VIMs managed by the MANO entity and the WIMs in between, we introduce a new component named NFV Coordinator (NFV-C) which is responsible of handling slice requests coming directly from Operation Support Systems (OSS) / Business Support Systems (BSS) or the 5G Vertical. Furthermore, it is also in charge of providing guarantees to maintain NSs according to agreed Key Performance Indicators (KPI) and Service Level Agreements (SLA).

The Slice Manager (SM) at the NFV-C, uses the Slice Composition technique to trigger the deployment of the slice across VIMs and WIMs. A more in depth look at this technique is given in the next subsection. The Tunnel Manager (TM) and the Interface Manager (IM) in turn, are responsible for setting the interconnection between deployed VNFs. To accomplish this, an overlay tunnel is configured from source VIM to destination VIM where each VNF resides. The process starts by requesting the controllers at the edge VIMs to add interface and tunnel data to their databases, so this can be configured at the underlying Data Centre Network (DCN). The Inter-VIM Manager (IVM) module has been designed and added at the SDN Controller level to trigger these configurations upon the request from the NFV-C. Once the edge VIMs are configured, the controllers at the WIMs are also requested to set up required connectivity between VIMs. After setting up the tunnel, the VNFs are able to reach each other, completing the service chaining between them.

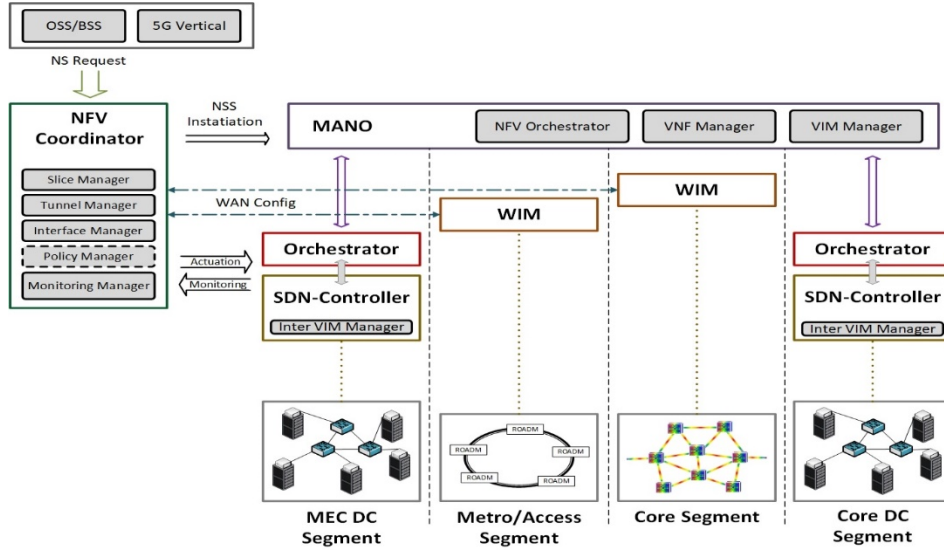


Fig. 1. Architecture for NS deployment and maintenance in optically interconnected DCs.

Regarding the maintenance of the slice, the Monitoring Manager (MM) gathers and processes data from the segments where the slice is deployed. Monitoring data can be

retrieved at different levels, directly from the VNFs, from the SDN controller or from the network orchestrator. It is important to notice that monitoring of a particular slice should be focused only in parameters relevant to maintain the KPIs/SLAs agreed for the slice. By processing this information, the MM can act preventively in case service guarantees are compromised. The Policy Manager (PM) is the one responsible for executing the required configuration changes to maintain the proper slice functionality. These configurations are recognized as actuations and can also be executed over different levels, through the network orchestrator, at the SDN controller or directly at the data plane. A policy-based approach is used in this case, where the Policy Manager applies specific policies according to the received monitoring metrics.

2.2 Slice Composition

As introduced in the previous subsection, the SM uses a technique for provisioning NSs across multiple network segments. The method in particular, follows the Slice Composition concept (i.e., “slice-cum-slice”) introduced by the 5G-PPP in [4]. It entails the construction of a slice from a set of individual slices. This means that a Network Slice Instance (NSI) can be instantiated by deploying a set of Network Slice Sub-Network Instances (NSSI) [5] and joining them together. Each NSSI represents configurations and resources deployed at one particular segment of the network. Following the architecture proposed in this work, upon the request for a NS, the SM decomposes it and asks the MANO entity for the required NSSI instantiation at specific VIMs according to resource and function availability. Moreover, it also contacts the TM and IM to trigger configuration at correspondent WIMs in between. After VNFs are instantiated and all configurations have been set across the segments, the NS becomes available for its utilization.

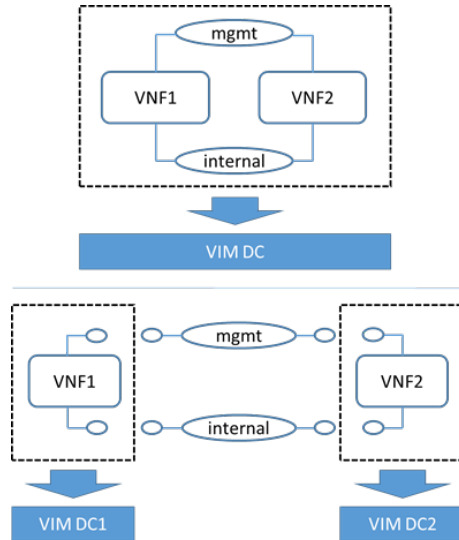


Fig. 2. NS provisioning through Slice Composition.

Fig. 2. depicts an example of how a NS containing 2 VNFs, can be de-composed and instantiated in different DC segments, thereby using 2 NSSI to compose the original slice, when their provisioning requirement constrains demand them to be allocated in different segments. In this regard, enabling communication between VNFs needs to be considered, so a coordination entity in charge of this process is required. Using this approach, an end-to-end (E2E) NS can be deployed across different segments of the network by instantiating a set of NSSIs and configuring connectivity between them. That said, VNFs contained in this E2E NS are chained together in a specific order following the basis of service-chaining [6].

3 LATENCY SENSING & ACTUATIONS

Considering the demanding requirements given for 5G services, latency awareness becomes crucial for network slices operation and maintenance. Especially when agreed KPIs/SLAs include a maximum latency value for the proper service operation. This section covers the proposal made in this work to perform latency sensing inside a particular NS to enable real-time reconfiguration (i.e. actuation) when latency measurements reach an undesired level.

3.1 Sensing Mechanism

The mechanism presented for latency measurement is based on the presence of a latency sensor in the service chain associated to the provisioned end-to-end service. More specifically, this sensing entity, deployed as a VNF, analyses the TCP traffic passing through the service chain and estimates the latency based on the time delay computed between a TCP packet and the associated ACK. Fig. 3 illustrates how the sensor VNF captures the packets exchanged by two end-points (VNFs as well in our case) and estimates the overall end-to-end latency introducing the minimum possible overhead. In this approach, Packet-LEFT arrival time (TPL) serves as a reference to calculate the round-trip time between the sensor and VNF2 (RTT-R). Therefore, after receiving and associating the ACK message for this particular packet, we use this PACKET-LEFT-ACK arrival time (TPLA) to the first part of the calculation. The operation will consider $RTT-R = (TPLA - TPL)$. Similarly, we can also calculate the other way, the round-trip time connection between the sensor and VNF1 (RTT-L), by using $RTT-L = (TPRA - TPR)$.

After having the results of both RTT-L and RTT-R, the next step will be to calculate the overall latency (L) between VNF1 and VNF2 with $L = ((RTT-L + RTT-R) / 2)$. The main responsibility of the sensor is taking samples of the traffic to perform these calculations during the slice lifetime and dumping the results to a local database so these become available via the management network to high layer clients (e.g., NFV Coordinator, Network Orchestrator).

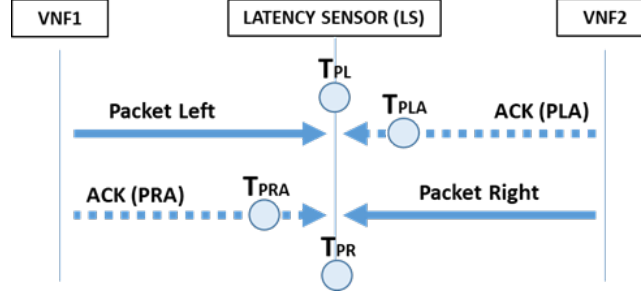


Fig. 3. Latency Sensing Mechanism.

3.2 Latency Sensor Provisioning

As introduced previously, the Latency Sensor (LS) considers having an entity deployed in the service chain associated to the slice. The approach entails that the LS VNF is provisioned during the NS provisioning stage. In this matter, the slice composition technique used in this work enables inserting the sensor in any of the decomposed NSSIs. As an example, if the NSSIs from Fig. 2 are considered, the LS can be deployed at the left NSSI as seen in Fig. 4.

The modified NSI will still consider the same VNF1 and VNF2 as originally, but in this case, the LS will be present as another VNF in between. The modified left NSSI will now contain VNF1 and LS connected via an internal network, besides their connection to the management network. Then, the LS will be the one to connect to VNF2 via the data network. In this sense, all work traffic flowing from VNF1 to VNF2 and back will pass through the LS, thus allowing it to capture samples of this traffic and execute its sensing mechanism.

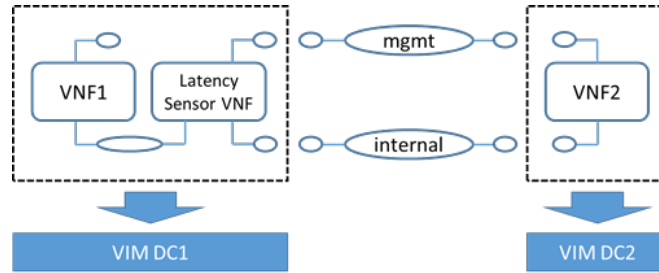


Fig. 4. Latency Sensor provisioning through Slice Composition.

3.3 Latency-aware Path Selection Algorithm

In addition to the abovementioned LS, the introduced NFV-C is equipped with a latency-aware path selection algorithm to select the most appropriate end-to-end path across the involved segments to be compliant with latency requirements of deployed NS. More specifically, the TM executes such algorithm when it has to determine the

route across the network segments interconnecting the DCs in which VNFs have been deployed.

In more details, the algorithm exploits the information gathered by multiple LSs. By extracting the measurements of the sensors deployed in each active NS, as well as the topological information exposed by network segment controllers, the algorithm correlates the route selection of previous deployed services with end-to-end measured latencies. Then, a graph representation of the multiple involved network segments and the estimated latency when crossing them following specific routes is constructed. With this, upon routing requests due to new service deployments or re-routing operations as consequence of actuations, the algorithm executes a simple shortest path mechanism, employing as weights of the constructed graph the estimated latencies, thus determining the sequence of segments and corresponding internal paths with the lowest latency.

3.4 Actuations over Latency Monitoring

After deploying the sensor along with the slice, it starts collecting latency monitoring data from work traffic and saving it in its local database. The MM at the NFV-C then begins to gather this information straight from the LS through the management network and proceeds to analyze it. By processing such data and comparing it to the pre-established thresholds desired for latency levels, the MM can react preventively upon the possibility of violating agreed service guarantees.

The approach taken in this work considers policy-based configurations (i.e., actuations) to allow maintaining latency levels across a slice. In this matter, an information model [7] provided by the Simplified Use of Policy Abstractions (SUPA) working group initiative at the IETF allows representing different policies according to the case. Such policies are set following the ECA model (i.e., event-condition-action) to define the reason for triggering the policy (e.g. high latency), the threshold to be surpassed (e.g., greater than 1ms) and the action to be executed (e.g., re-route slice traffic) to guarantee the maintenance of the slice according to agreed KPIs/SLAs.

4 EXPERIMENTAL TESTING

In order to test the whole provisioning and runtime maintenance of a particular slice, an experimental multi-segment testbed (illustrated in Fig. 5) has been set up. Starting at the data plane, the testbed is composed of four emulated network segments: the MEC DC (left), the Metro/Access Network (middle), the Optical Core Network (middle) and the Core DC (right). Network resources at these particular segments are emulated using Mininet networking tool [8] to allow connecting computing resources from edge to edge. In the case of MEC and Core DCs, an Opendaylight (ODL) [9] SDN controller in its Carbon release provides control of both DC networks. Moreover, OpenStack [10] orchestrators at Queens release are also present at these segments to manage computing resources.

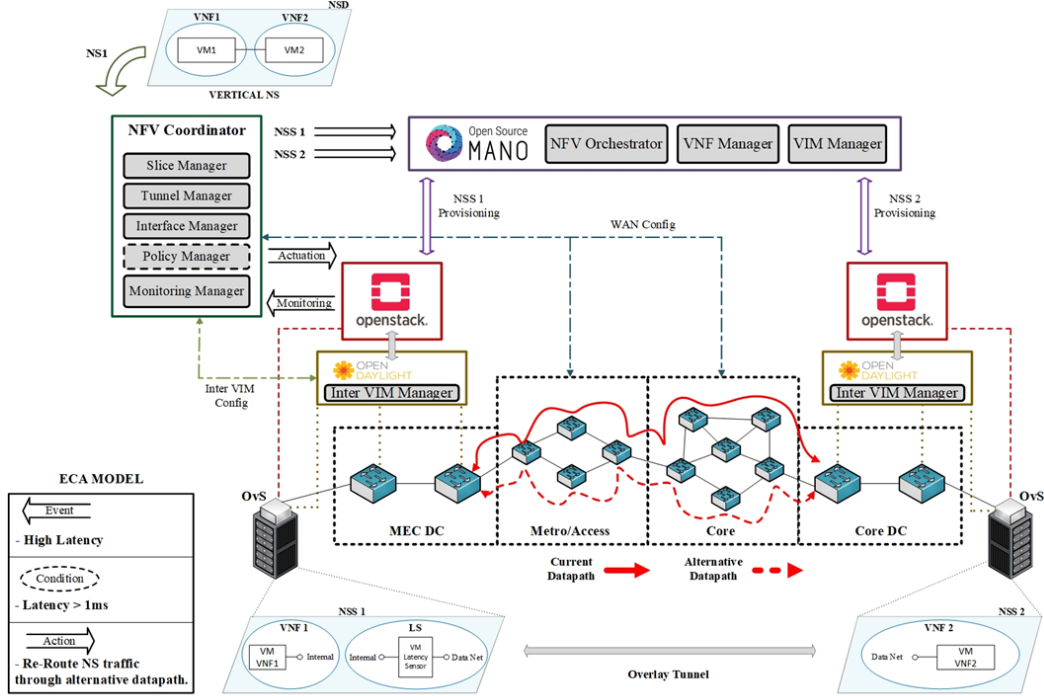


Fig. 5. Experimental multi-segment testbed for NS provisioning and maintenance.

For our purposes, to allow for the configuration of optical resources at SDN controlled network segments, an extended version of ODL has been used, implementing extensions both at the southbound protocol and SDN applications levels [11]. The IM and TM components at the NFV-C in turn, upon request from the SM, configure the optical paths of the Metro/Access and Core networks, which provide the physical connectivity to the virtual IP tunnel that interconnects the service chain VNFs.

Above the segments control and orchestration, an instance of the Open Source MANO (OSM) [12] framework in its fourth release enables VIM registration and NSSI deployment over the OpenStack enabled segments. OSM has direct interface with the SM, which triggers NSSI requests to OSM at the slice provisioning stage. The NFV-C component (depicted in the upper left side of Fig. 5) contains the SM, IM, TM, PM and MM components previously discussed in Section 2A. In practice, the interaction between these components and the open source frameworks deployed in the testbed may use different channels. In some cases, as the SM-OSM interaction, it runs through the OSM REST northbound interface (NBI). The IM-ODL connection on the other hand, uses ODL REST API and the remote procedure calls (RPC) defined at the IVM. Moreover, other interactions such as the one from TM-Mininet at the intermediate segments and the one from MM-VNFs are set through a series of scripts.

The slice provisioning stage begins upon the arrival of a network service request from the client (e.g., OSS/BSS, 5G Vertical), based on a Network Service Descriptor (NSD). In the realized tests, the NSD representing the virtual infrastructure is composed of two client VNFs (VNF1 and VNF2) and the latency sensor (LS) lying in between as part of the service chain. Then first, the SM decomposes the NSD into two NSSIs, as depicted in Fig. 2. The next step entails the request from SM to OSM for the deployment of NSSI-1 (VNF1 and LS) and NSSI-2 (VNF2) over OpenStack-LEFT and OpenStack-RIGHT VIMs respectively. While OSM contacts both VIMs for compute instantiation and IP network configuration, the TM configures the physical data path that connects the DC segments (i.e. the optical path crossing the Metro and Core segments) according to a simple path-selection routing algorithm, which considers latency as detailed in Section III-C.

As soon as each DC VIM configures the local VNF network interface, it notifies the IM at the NFV-C, which triggers the cross-configuration of each interface at the other side DC VIM using RPCs defined by the IVM component. The main idea is that the Open Virtual Switch (OVS), which connects VNF1 and LS, is configured by ODL to flow traffic going to VNF2 through the established overlay tunnel over the Metro and Core network segments. This configuration is also set in the opposite way, allowing work traffic to flow over the data network, thus completing the VNF1-LS-VNF2 service chain. After triggering and executing slice configurations from SM, TM and IM via OSM, ODL and OpenStack, the NSI including all of its components becomes operative, thus concluding the provisioning stage. Fig. 6 shows the OSM dashboard (a) with both NSSI-1 and NSSI-2 configured and running over the registered openstack-left and openstack-right VIMs. The OpenStack dashboard from the MEC DC (b) then shows configured VNF1 and LS, while the one from Core DC (c) shows VNF2. As seen in the figure, VNF1 connects to LS via an internal network, and then LS connects to VNF2 through an overlay IP tunnel configured over the data network. Finally, all VMs/VNFs also connect to the provider/management network for external access.

With the NSI up and running, the runtime stage, where MM and PM are the ones in charge of the slice maintenance, begins. In this case, the LS starts to capture samples of work traffic between VNF1-VNF2, computes the latency estimation, and dumps it to its local repository, which then accessed by the MM. After analyzing the data and upon the case of compromised latency levels, the MM contacts the PM to execute pre-established policy-based actuations following the ECA model. Actuations in this particular case, consider reaching the path-selection routing algorithm in order to re-route the physical data path associated to the virtual IP connection of the specific service. As said before, the path-selection algorithm uses overall latency information to compute a new data path fulfilling service latency requirements. Fig. 5 illustrates the validation of the architecture. We saturate the current data path at the WAN segments by creating traffic congestion, in a way to surpass the permitted latency levels. In this way, the policies are activated and the overlay tunnel IP traffic is routed through an alternative data path. By means of this exercise it is possible to complete the flow for monitoring and policy-based actuations.

The results achieved using this testbed, prove the expected behavior for the whole architecture in both the provisioning and maintenance stages of the NSI. As for the future work, the NFV-C component still requires enhancement to accommodate its components to other functionalities. In this regard, the SM may also enable triggering NSSI deployment over different MANO entities according to the scenario. Furthermore, the MM and PM should be extended to provide slice maintenance according to other parameters, such as Bit Error Rate (BER), CPU usage, RAM usage, etc., as well as considering the implementation of cognitive-based predictive mechanisms. Finally, the TM to WIM interaction should also be furtherly enhanced.

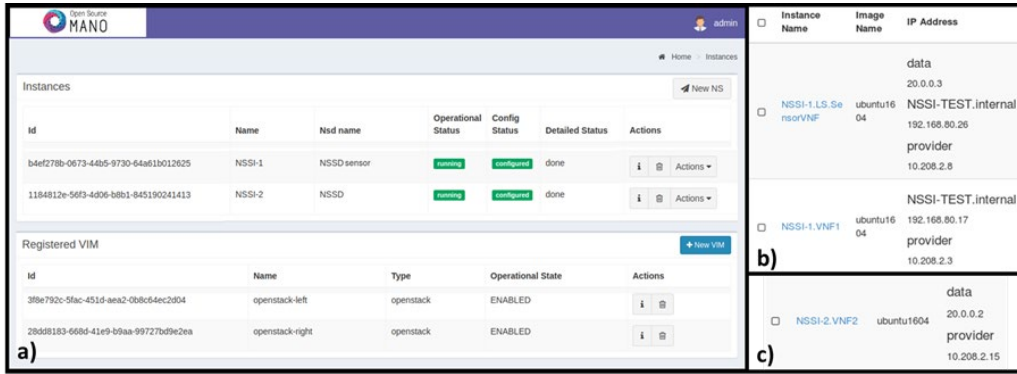


Fig. 6. Open Source MANO (a) and OpenStack (b, c) dashboards with deployed NSI.

5 CONCLUSIONS

With the struggle to accommodate legacy network scenarios to the demanding requirements of modern networks and 5G services, there rises a need to have more up-to-date information regarding network operation. In this matter, we presented in this work a method to address latency sensing in multi-segment scenarios prepared for the deployment of network slices.

The architecture presented in this work allows coordinating VIM and WIM network segments for the provisioning and maintenance of network slices. A particular NFV-C component has also been introduced to work as the core part of this architecture, providing a common point for communication between the network client, the MANO entity and other network components/resources. Then, an experimental testbed has been used to demonstrate both the deployment of a NSI by means of the slice composition technique, while also considering its ongoing maintenance in terms of contracted latency levels. In this regard, experimental validation shows the correct NSI to NSSI-1 and NSSI-2 decomposition and instantiation over DC segments connected via an emulated WAN network. Moreover, it provides an example of latency monitoring and actuation, where upon the violation of a pre-established condition, a re-configuration of the routing path is triggered.

ACKNOWLEDGEMENT

This work has been supported by the H2020 5GPPP SLICENET project (H2020-ICT-2016-2/761913) and the Spanish Government through project ALLIANCE-B (TEC2017-90034-C2-2-R) with FEDER contribution.

REFERENCES

1. F. J. Moreno-Muro et al., "Latency-Aware Optimization of Service Chain Allocation with Joint VNF Instantiation and SDN Metro Network Control," 2018 European Conference on Optical Communication (ECOC), Rome (Italy), 2018, pp. 1-3.
2. R. Montero et al., "Supporting QoE/QoS-aware end-to-end network slicing in future 5G-enabled optical networks", PW 2019, San Francisco (United States), 2-7 February 2019.
3. ETSI GS NFV-MAN 001 V1.1.1, December 2014.
4. 5G-PPP 5G Architecture White Paper, Version 2.0, December 2017.
5. 3GPP TR 28.801, Study on management and orchestration of network slicing for next generation network, Version 15.1.0, January 2018.
6. G. Herrera and J. F. Botero, "Resource Allocation in NFV: A Comprehensive Survey," in IEEE Transactions on Network and Service Management, vol. 13, no. 3, pp. 518-532, September 2016.
7. IETF draft-ietf-supra-generic-policy-info-model-03: "Generic Policy Information Model for Simplified Use of Policy Abstractions (SUPA)", May 30, 2017.
8. Mininet, <https://mininet.org>.
9. OpenDaylight, <https://www.opendaylight.org>.
10. OpenStack, <https://www.openstack.org>.
11. S. Spadaro et al., "Resource orchestration in SDN-based future optical data centres," 2016 International Conference on Optical Network Design and Modeling (ONDM), Cartagena (Spain), 2016, pp. 1-6.
12. Open Source MANO, <https://osm.etsi.org>.