Towards Cross-Slice Communication for Enhanced Service Delivery at the Network Edge

Ioakeim Fotoglou, George Papathanail, Angelos Pentelas, Panagiotis Papadimitriou
Department of Applied Informatics, University of Macedonia, Greece
{fotoglou, papathanail, apentelas, papadimitriou}@uom.edu.gr

Vasileios Theodorou
Intracom Telecom, Greece
{theovas@intracom-telecom.com}

Dimitrios Dechouniotis, Symeon Papavassiliou
National Technical University of Athens, Greece
{ddechou@netmode, papavas@mail}.ntua.gr

***Abstract*—The increasing resource demand and diversity of network services is taken under serious consideration by the various stakeholders, driving the architecture design of 5G (and beyond) networks. Network slicing, as a prominent aspect of next-generation network architectures, aims at satisfying the diverse service requirements in terms of throughput, latency, reliability, and/or security. However, the prevailing way of slice provisioning, *i.e.,* in the form of isolated bundles of computing, storage, and network resources, makes cross-slice communication inefficient, especially at the network edge. This inevitably hinders opportunities for Business-to-Business (B2B) synergies at the event of service co-location.**

In this paper, we study this novel aspect of network slicing, *i.e.,* cross-slice communication (CSC). We particularly promote a form of optimized CSC, at which two co-located slices can establish peering in a secure and controlled manner, by confining peering traffic within the boundaries of the datacenter, while still preserving the important aspect of resource isolation. Such optimized CSC can foster synergies between service providers without additional latency or traffic in the backhaul/transport network. In this context, we investigate various ways to establish optimized CSC at edge computing infrastructures, based on functionalities offered by state-of-the-art management and orchestration (MANO) frameworks, such as OpenSourceMANO.

I. Introduction

We stand at the dawn of the fifth generation (5G) [1] networks era, which will change the way we manage, operate, and conceive networks in the near future. Several research projects investigate how 5G networks can satisfy service demands, *e.g.,* high throughput, ultra-low latency, as well as increased reliability that consumer and business services require. A significant driving force for the aforementioned requirements is the increase in the diversity of provided services, often related to various domains of social and economic activity, termed as verticals (*e.g.,* media, healthcare, energy, automotive).

A network which acts as a general carrier for all sorts of services can no longer meet these strict and diverse service demands; as such, a complete reconceptualization of network architecture, backed up by network programmability and softwarization, is needed. Prominent examples of such enablers are Network Function Virtualization (NFV) [2]–[4] and Software Defined Networking (SDN) [5]. Based upon the principles of NFV and SDN, network slicing [6], [7] is constantly gaining ground in many application domains, such as telco networks, clouds, and especially in the emerging 5G cellular networks. Network slicing enables infrastructure providers to lease service-tailored logical networks, in the form of isolated bundles of compute, storage, and network resources. This can essentially enable the concurrent deployment of services with potentially different requirements, which may be expressed in terms of KPIs or resource demands.

Network slicing commonly mandates strong isolation (between different slices), inhibiting cross-slice communication (CSC), even if the slices are located in the same datacenter, rack or server. While complete slice isolation is dictated due to security and performance concerns, the inability of co-located slices to communicate increases the latency, as well as the amount of traffic injected in the backhaul and the transport network. More specifically, the prevailing slicing approach enforces all CSC traffic to exit the boundaries of the datacenter and traverse all the way to the core network, before re-entering the datacenter. This yields a higher penalty in terms of delay and bandwidth consumption in the transport network for edge computing, since the associated infrastructures (*i.e.,* micro-datacenters) are usually not proximate to the (mobile) network core. This is expected to degrade the performance of latency-sensitive and bandwidth-intensive services that require cross-slice interactions. Along these lines, we promote optimized CSC, *i.e.,* the establishment of peering between two co-located slices in a secure and controlled manner, where the entire peering traffic is confined within the boundaries of the datacenter [8].

In this paper, we investigate practical ways for the estab-

978-1-7281-5684-2/20/$31.00 ©2020 IEEE

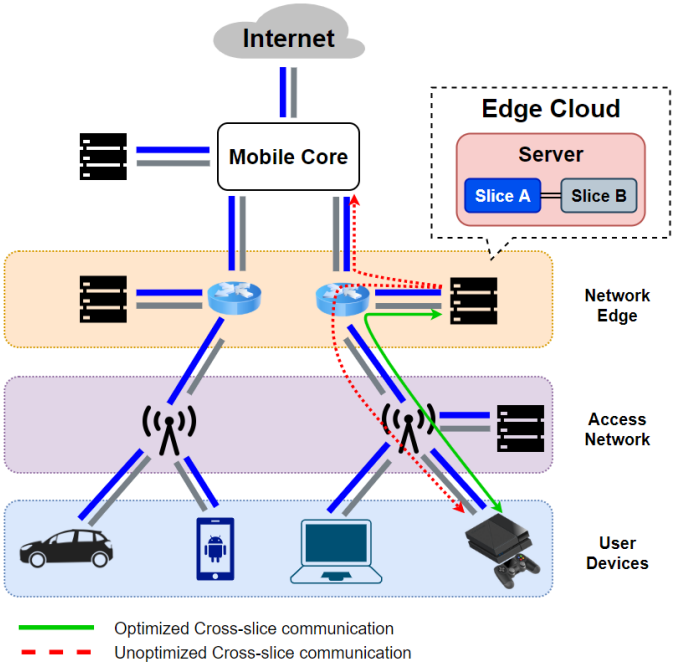

Fig. 1. Optimized vs. un-optimized cross-slice communication.

lishment of CSC among co-located slices in the context of edge computing, where the gains from optimized CSC are more prominent. In this respect, we rely on the functionalities provided by state-of-the-art management and orchestration (MANO) platforms, such as OpenStack [9] and OpenSource-MANO (OSM) [10]. In particular, CSC is instantiated via a dedicated slice, acting as an intermediate arbiter between the two communicating slices. This CSC-slice, managed by the infrastructure provider, essentially constitutes a secure, controlled, and optimized communication path that facilitates cross-service interactions and B2B synergies. In terms of CSC setup, we experimentally investigate two different scenarios, both of which are based on the notion of the intermediate CSC-slice: (i) a baseline CSC scenario, at which the CSC-slice encompasses dedicated services, and (ii) a shared CSC scenario, at which one or multiple services are shared among CSCs. These services can include functions (which are expected to be realized in the form of virtualized network functions - VNFs), such as monitoring and packet inspection. Based on these scenarios, we measure the time required for the instantiation of CSC slices and show that CSC can be established in the order of seconds. We further demonstrate a notable reduction in the instantiation time when binding running VNF instances to a CSC-slice.

The remainder of the paper is organized as follows. In Section II, we discuss the concept and the expected gains from optimized CSC. Section III provides an overview of MANO. In Section IV, we discuss two CSC scenarios, whereas in Section V, we compare the performance achieved with these scenarios, using an experimental setup. Related work is presented in Section VI. Finally, in Section VII we highlight

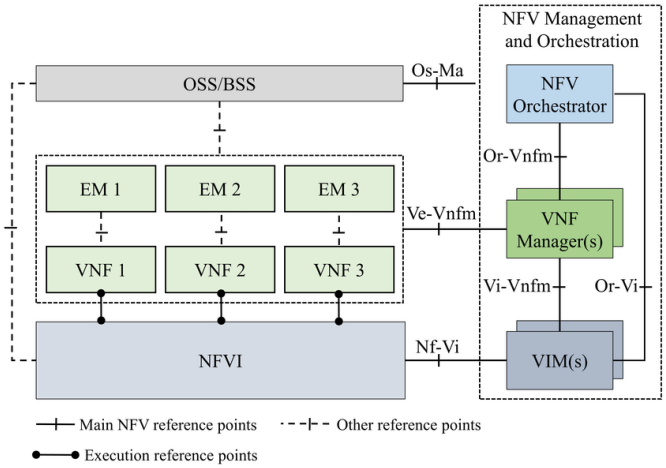

Fig. 2. MANO architecture overview.

our conclusions.

II. Cross-Slice Communication

The prevailing network slicing mechanisms enforce strong resource isolation among co-located network slices mainly for security reasons. This restriction introduces significant limitations to cross-slice communication, from which service providers can greatly benefit in potential cases of service co-location. For instance, an augmented reality service in a smart city environment can benefit from a co-located social network service that can provide personalized recommendations. Additional use cases for CSC are described in [8].

Network slice isolation, especially in edge computing infrastructures, hinders such opportunities for cross-service interactions. This limitation is illustrated in Fig. 1. In particular, the red dotted line represents the communication path between two co-located (but isolated) slices. Although these slices are located in the same infrastructure (or even in the same rack or server), routing policies (stemming from the need for resource/traffic isolation) will direct the traffic from one slice outside of the datacenter, through the core network, and back to the same datacenter in order to reach the other slice. Ideally, CSC traffic should be confined within the same datacenter by setting up a form of direct peering between the two slices. This so-called optimized CSC is depicted with the green line in Fig. 1. We note that such optimized CSC can be attained in a secure and controlled manner, without compromising the isolation between the slices in the (edge) cloud infrastructure.

The gains from optimized CSC are numerous. First of all, user experience from the consumption of a service will be enhanced, as the effect of minimal latency over the optimized CSC path. This may further generate increased revenue for Service Providers, since they will be in a position to expand their customer base, due to the potentially higher QoS compared to other similar service offerings. These gains will be more evident for latency-sensitive services, such as location-based services. At the same time, the Service Provider will benefit from reduced expenditure, since optimized CSC will

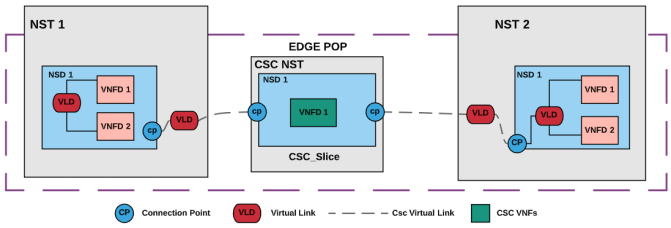

Fig. 3. Baseline CSC scenario.

obviate the need to purchase additional bandwidth from the backhaul and transport network. This will be very critical for bandwidth-intensive services, which become more prevalent in the 5G era. As a consequence, optimized CSC is expected to reduce the traffic load in the backhaul and transport network.

As CSC fosters interactions between two Service Providers, we consider the following two roles: (i) a CSC-enabled Service Provider, who offers a service that is deployed on a network slice via CSC, and (ii) a CSC-enabled Service Consumer, who wishes to consume a service from a co-located slice via CSC. Mechanisms for expressing CSC intents and discovering offered CSC-enabled services are currently under investigation and will be developed and evaluated in future work. The focus of this paper is to explore ways for the establishment of optimized CSC between two co-located slices within the same datacenter. To this end, we rely on MANO and its associated descriptors (briefly described in Section III).

To establish CSC, we use a dedicated intermediate slice (instantiated and managed by the infrastructure provider), which provides (i) connectivity between the two communicating slices, in a secure and controlled manner, and (ii) selected management, monitoring, and security functions, required either by the CSC-enabled Service Provider (*e.g.,* policy control, security, billing, etc.) or the CSC-enabled Service Consumer (*e.g.,* monitoring). These functions can be deployed within the CSC slice in the form of a service chain. In Section IV, we exemplify the establishment of CSC with shared functions (*i.e.,* binding running function instances to a new CSC slice), with the aim to accelerate the CSC setup, as well as to reduce the management overhead for the infrastructure provider.

III. MANO Background

Since the establishment of optimized CSC is based on NFV MANO platforms, *i.e.,* OpenStack and OSM, we hereby provide background information on the MANO architectural framework (Fig. 2) and the associated descriptors. MANO encompasses two main layers. The top layer is responsible for service and resource orchestration. The MANO orchestrator has an abstract view of the entire network and provides orchestration functionalities, such as VNF lifecycle management. The bottom layer, known as the *Virtual Infrastructure Manager* (VIM), handles the resource (*i.e.*, compute, storage, network) management of the underlying physical infrastructure.

MANO utilizes entities in order to create (i) VNFs, (ii) Network Services, and (iii) Network Slices. These entities are called descriptors, which essentially comprise configuration templates defined in YAML format. The lowest structural deployment unit provided by MANO is the VNF, which is defined by the *Virtual Network Function Descriptor* (VNFD). A VNFD includes information, such as the resources that the corresponding VNF requires. In MANO, a Network Service is defined as a sequence of VNFs. To this end, MANO utilizes a *Network Service Descriptor* (NSD), which specifies the network service graph, *i.e.,* the constituent VNFs along with their virtual links. Finally, MANO also allows the specification of a network slice using the *Network Slice Template* (NST), which is the top-level entity that MANO can orchestrate. The NST describes inter-connected Network Services including their network elements as virtual links. In addition, NST allows the definition of various attributes, such as specific service types, *e.g.*, enhanced MobileBroadband (eMBB), ultra-reliable low latency communication (URLLC) and massive IoT (MIoT).

IV. CSC Scenarios

In this section, we elaborate on CSC through the detailed examination of two scenarios that reflect different approaches to CSC. As already mentioned, our CSC setup relies on MANO and the descriptors defined by this NFV orchestration architecture. More precisely, network slices are defined using NSTs, each containing one or multiple NSDs. In turn, the NSDs are associated with VNFs which are defined using VNFDs. Although a NSD can include multiple VNFDs, in our setup, each NSD associated with the intermediate CSC slice contains only a single VNFD.

The two main CSC scenarios under our consideration are the following:

Baseline CSC scenario. Fig. 3 illustrates the baseline scenario for CSC instantiation. We assume that two slices from different tenants, along with the services they deploy, are up and running in the same edge cloud infrastructure. Optimized communication between these two slices is established via an intermediate dedicated slice, as described in Section II. More specifically, the intermediate CSC slice is an independent slice owned and operated by the infrastructure provider. Besides communication, a CSC slice can contain management and security functions, in the form of VNFs, used by the communicating slice tenants. We point out that, unlike the communicating slices which may include segments in other infrastructures, the intermediate slice is confined within the edge cloud infrastructure that CSC is being set up. A particular restriction introduced by this scenario is that each VNF is dedicated to a single CSC slice, and as a consequence, to a specific pair of communicating slices.

Shared CSC scenario., We point out that edge infrastructures are commonly resource-restricted environments and, as such, an note-worthy aspect is the sharing of CSC management and security functions among different CSCs. In such scenario, a pool of CSC management NSDs can be pre-configured and instantiated by the corresponding infrastructure provider, facilitating their binding with CSC slices, during their instantiation.

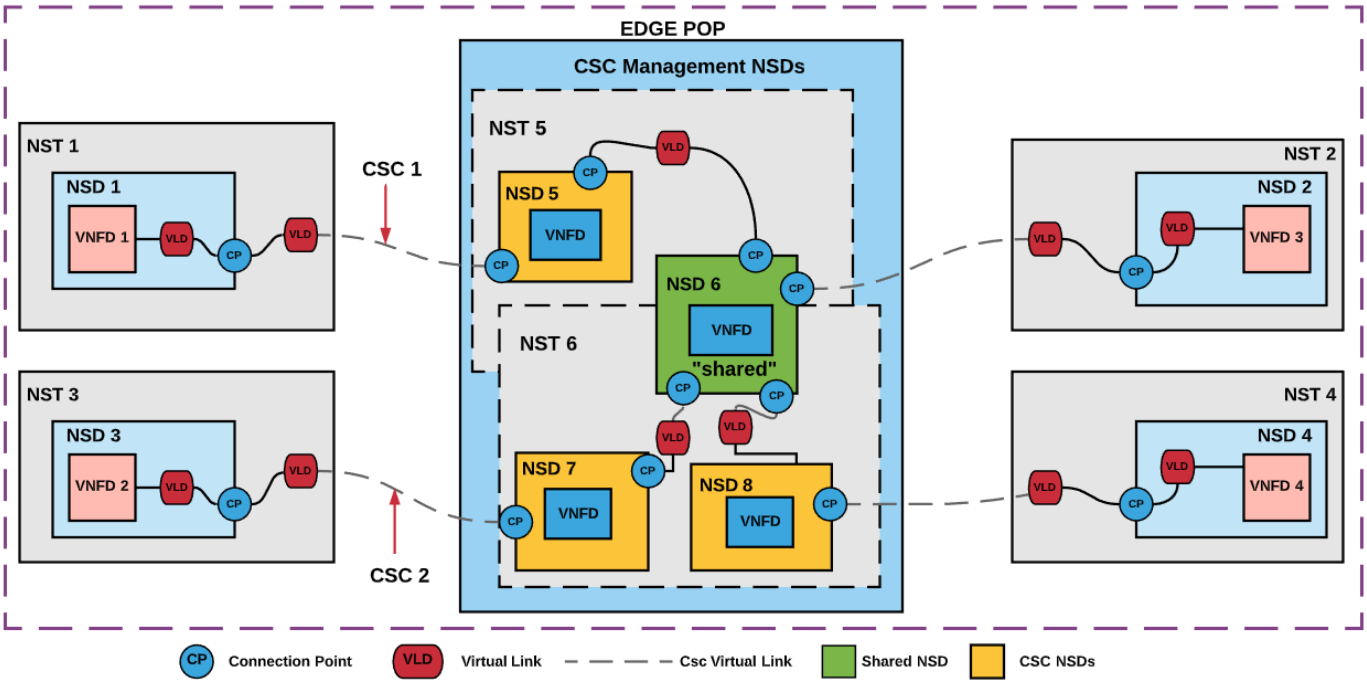

Fig. 4. Shared CSC scenario.

As such, CSC establishment can be accelerated, since the need for spawning new VNF instances is obviated. In essence, CSC setup mainly requires network configurations and service chaining. Nevertheless, careful attention should be paid by the infrastructure provider at the CSC management level, possibly with additional configurations (*e.g.*, VNF state management, policing), to ensure an appropriate level of isolation between pairs of slices.

Fig. 4 illustrates such a shared CSC. In particular, two pairs of communicating slices (*i.e.,* NST 1–2 and NST 3–4) share NSD 6, while NSD 5 is dedicated to CSC 1 (connecting NST 1 with NST 2) and NSDs 7–8 are associated with CSC 2 (connecting NST 3 with NST 4). As already explained, the sharing of NSD 6 can lead to resource savings for the infrastructure provider, as well as faster CSC instantiation.

V. Evaluation

In this section, we discuss the experimental evaluation of the baseline CSC and the shared CSC setup. We initially present our experimental setup (Section V-A) and, subsequently, discuss our experimental results (Section V-B).

A. Experimental Setup

All tests are conducted on a Dell PowerEdge R440 server with a single Intel Xeon Silver 4110 CPU with 8 CPU cores at 2.1 GHz and 128 GB of RAM. The server hosts OSM version 6.0 and OpenStack (which acts as the VIM). OpenStack manages three virtualized compute nodes deployed in VMWare. Each of these nodes has allocated 8 VCPUs, 32 GB of main memory, and 100 GB of virtual storage on the same physical disk.

In the following, we discuss our experimental setup for the baseline and the shared CSC scenario. Fig. 5 illustrates the VNF chain, deployed in the intermediate slice for the baseline scenario. The service chain consists of three VNFs for security and management of the ingress/egress traffic. More specifically, we employ Vuurmuur [11] for the firewall, we utilize the open-source Suricata [12], capable of real-time intrusion detection and network security monitoring, and also use Monitorix [13], which is an open-source tool for monitoring services and system resources. All VNFs utilize the same amount of resources *i.e.*, 2 vCPUs, 2GB RAM, and 20GB of storage.

For the shared CSC scenario, we consider two CSCs (*i.e.,* between two pairs of CSC-enabled Consumers and Providers). Each CSC is established through an individual CSC slice. However, unlike the baseline scenario, we permit the sharing of selected NSDs between the two (intermediate) CSC slices. The shared NSD, which, in our case, is associated with a single VNF, is reused by the second CSC slice, obviating the need for the deployment of another VNF instance. As such, there is a great potential for both instantiation delay and resource savings, since services of high-demand can be provisioned with a smaller number of instances. Although in our tests we utilized the same service chain (*i.e.,* firewall-IDS-monitoring) for both CSCs, VNF sharing may be as well enabled among different service chains that have common VNFs. For our experimental purposes, we run tests with two variants of the shared CSC scenario: (i) one VNF (*i.e.,* IDS) shared among the two CSC slices, and (ii) two VNFs (*i.e.,* firewall, IDS) shared among the two CSC slices.

In order to compare the performance between these CSC

scenarios, we measure the delay incurred till the CSC slice has been instantiated and the peering between the two communicating slices is established and fully functional. This essentially includes the time spent for VNF instantiation, chaining, and the bridging of the CSC slice with the two communicating slices. The reported CSC instantiation times are obtained through parsing the instantiation logs from the VIM. Instantiation times are measured from the moment a CSC request is initiated, till all required VNF instances have been built, chained, and bridged to the slice of the Provider and the Consumer. All tests are executed across 30 runs, for each CSC scenario.

B. Experimental Results

We hereby discuss the measured CSC instantiation times for the various CSC scenarios. Besides CSC instantiation, we also present various statistics to gain more insights from our experimentation. Fig. 6 illustrates box plots for the (i) baseline (non-sharing) scenario, (ii) 1 shared VNF (Sharing 1), and (iii) 2 shared VNFs (Sharing 2). As shown in this plot, VNF sharing yields significantly lower instantiation time, since the respective shared VNFs are already deployed; hence, only a subset of the service chain needs to be instantiated. We further observe that the span in the measured times is shorter in the two variants of shared CSC. This is attributed to the following two factors: (i) the reduction of the instantiation time, since fewer VNFs need to be instantiated in the shared scenarios, and (ii) the reduction of instantiation times observed in cases where an already existing VNF is re-spawned in the VIM.

With respect to the latter, we note that individual instantiation times for each VNF appear to be reduced by up to 24% in the case of the monitoring VNF, compared to non-sharing. This stems from image caching and/or reusing by the VIM[1] and is not a factor pertaining directly to the CSC. This issue has a significant impact, when comparing any of the two shared CSC variants with the baseline scenario. On the other hand, image caching is not expected to make a difference between the two shared CSC variants, since all images are already instantiated in the first CSC slice, and thus, no gains can be directly attributed to image caching. As such, the gap in the instantiation time between one and two shared VNFs is lower (compared to baseline vs. sharing), as shown in Table I.

The time required for bridging is sampled and found to have a minimal contribution in the the total instantiation time, with a minimal standard deviation. Given that VNF instance spawning comprises the dominant factor for CSC setup, we expect lower instantiation times with VNFs assigned to compute nodes hosted on different servers. Overall, our experimental results indicate that CSC can be established in the order of seconds, with the potential to further reduce this delay through VNF sharing among CSC slices.

[1]https://docs.openstack.org/nova/latest/admin/image-caching.html

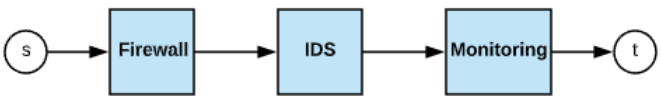

Fig. 5. Service chain deployed in the CSC slice.

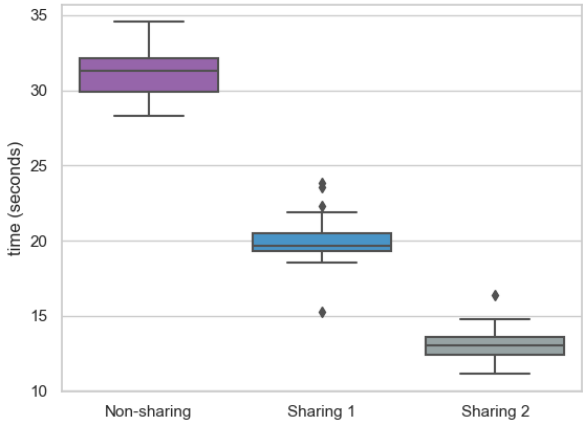

Fig. 6. CSC instantiation time for the baseline (non-sharing) scenario, the scenario with 1 shared VNF (sharing 1), and the scenario with 2 shared VNFs (sharing 2).

TABLE I
STATISTICS OF CSC INSTANTIATION

Non-Shared Scenario:

	Firewall	**IDS**	**Monitoring**	**Duration**
Average (sec)	17.34	17.28	16.21	31.26
Standard Deviation (sec)	1.95	1.38	1.44	1.70
MAX (sec)	21.84	19.54	20.32	34.54
MIN (sec)	14.55	14.59	14.22	28.32

Shared Scenario 1:

	Firewall	**IDS**	**Monitoring**	**Duration**
Average (sec)	15.64	0	14.63	20.06
Standard Deviation (sec)	1.72	0	1.14	1.59
MAX (sec)	20.46	0	18.18	23.81
MIN (sec)	13.90	0	12.84	15.23

Shared Scenario 2:

	Firewall	**IDS**	**Monitoring**	**Duration**
Average (sec)	0	0	13.04	13.04
Standard Deviation (sec)	0	0	1.15	1.15
MAX (sec)	0	0	16.39	16.39
MIN (sec)	0	0	11.12	11.12

VI. RELATED WORK

In this section, we provide an overview of related work on network slicing. Zhang *et al.* [14] present a logical Network Slicing architecture that is based on 5G systems, as well as a mobility scheme for managing different access networks. In [15], Simon *et al.* propose the use of SDN and NFV in a flexible 5G network architecture to enable rapid service deployment, automatic resource orchestration and sharing for an assortment of non-similar services. These "resource slices" can be conceptualized as virtual resources and VNFs and,

together being customizable, to accommodate different service categories and to be offered as a slice as a service (SlaaS) bundle. The concept of hierarchical Network Slicing as a Service (NSaaS) is presented in [16]. NSaaS enables operators to offer end-to-end cellular networks as a service to their tenants.

In [17], Vassilaras *et al.* discuss algorithmic aspects of network slicing. More specifically, they define network slice embedding, *i.e.*, the process of placing the slice parts onto candidate infrastructures, as a combinatorial optimization problem. The authors argue that this problem shares certain similarities with the well-known virtual network embedding problem, thus allowing the application of algorithmic methods of the former to the latter. The NECOS project [18] introduces a marketplace-based approach for a more pragmatic formulation on assigning the various slice ssegments across multiple physical infrastructures. Li *et al.*, in [19], investigate the application of a Deep Q-Learning (DQL) algorithm, which belongs in the family of Deep Reinforcement Learning (DRL) algorithms, for demand-aware resource allocation in different slicing scenarios. A comparative evaluation of their approach with a demand prediction-based algorithm, as well as some other intuitive methods, corroborates the benefits of the DQL approach.

Network slicing has been also studied in the context of cellular networks. One objective pursued by various studies is to optimize the placement of VNFs that implement certain elements of a virtualized cellular network, such as the eNodeB (eNB), the Serving Gateway (S-GW), the Packet Data Network Gateway (P-GW), and the Mobility Management Entity (MME). In this respect, authors in [20] propose an exact method for the embedding of such VNF-graphs onto a virtualized cellular core. Besides capacity constraints, the proposed method optimizes the placement of VNFs, taking into account delay budgets between VNFs, as mandated by 3GPP. *Papagianni et al.* [21] follow a different approach to this optimization problem, as they promote the sharing of VNFs among multiple service chains in a network slice, as means to reduce the deployed VNF instances and, consequently, the provisioning and management cost of the virtualized cellular network. To this end, the authors propose a MILP formulation for network slice embedding with shared VNFs.

In our work, we focus on the aspect of cross-slice communication, which has not received significant attention, despite its great scope for cross-service interactions and B2B synergies. In this respect, we have experimentally investigated the feasibility of two CSC scenarios using MANO platforms and measured the delay incurred during the asynchronous instantiation of CSC with both scenarios.

VII. Conclusions and Future Work

In this paper, we studied CSC in an attempt to facilitate interactions between services deployed in slices co-located in edge cloud infrastructures. Given the increasing interest in network slicing, we anticipate that optimized CSC will evolve into a prominent slicing orchestration aspect. In this context, we conducted initial performance tests, measuring the timescales at which CSC can be established, under different CSC scenarios that can be potentially mapped to resource allocation policies exercised by infrastructure providers (which are responsible for setting up CSC).

Our experimental results indicate the feasibility of CSC establishment, even with the baseline scenario that inhibits VNF sharing among the CSC slices. Besides the reasonable measured instantiation times, our experimental setup indicates that CSC can be established using MANO descriptors and orchestration mechanisms available in OSM and OpenStack. Nevertheless, the discovery of CSC-enabled services, as well as the expression of CSC intents requires additional functionality, not available in state-of-the-art MANO platforms. This is, in fact, our main direction of future work, within the scope of optimized CSC.

VIII. Acknowledgments

This work is supported by the MESON (Optimized Edge Slice Orchestration) project, co-financed by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH–CREATE–INNOVATE (project code: T1EDK-02947).

References

[1] N. Alliance, "5g white paper," *Next generation mobile networks, white paper*, vol. 1, 2015.

[2] "ETSI Network Function Virtualization," http://www.etsi.org/technologies-clusters/technologies/nfv.

[3] M.-A. Kourtis, M. J. McGrath, G. Gardikis, G. Xilouris, V. Riccobene, P. Papadimitriou, E. Trouva, F. Liberati, M. Trubian, J. Batallé *et al.*, "T-nova: An open-source mano stack for nfv infrastructures," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 586–602, 2017.

[4] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications surveys & tutorials*, vol. 18, no. 1, pp. 236–262, 2015.

[5] S. Agarwal, M. Kodialam, and T. Lakshman, "Traffic engineering in software defined networks," in *IEEE INFOCOM*, 2013, pp. 2211–2219.

[6] N. Alliance, "Description of network slicing concept," *NGMN 5G P*, vol. 1, p. 1, 2016.

[7] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5g: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[8] K. Katsaros, V. Glykantzis, P. Papadimitriou, G. Papathanail, D. Dechouniotis, and S. Papavassiliou, "Meson: Facilitating cross-slice communications for enhanced service delivery at the edge," 2019.

[9] "Openstack," http://www.openstack.org.

[10] "OSM," http://osm.etsi.org.

[11] "Vuurmuur," https://www.vuurmuur.org/trac/.

[12] "Suricata," https://suricata-ids.org/.

[13] "Monitorix," https://www.monitorix.org/.

[14] H. Zhang, N. Liu, X. Chu, K. Long, A.-H. Aghvami, and V. C. Leung, "Network slicing based 5g and future mobile networks: mobility, resource management, and challenges," *IEEE communications magazine*, vol. 55, no. 8, pp. 138–145, 2017.

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

[16] X. Zhou, R. Li, T. Chen, and H. Zhang, "Network slicing as a service: enabling enterprises' own software-defined cellular networks," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 146–153, 2016.

[17] S. Vassilaras, L. Gkatzikis, N. Liakopoulos, I. N. Stiakogiannakis, M. Qi, L. Shi, L. Liu, M. Debbah, and G. S. Paschos, "The algorithmic aspects of network slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 112–119, 2017.

[18] P. D. Maciel, F. L. Verdi, P. Valsamas, I. Sakellariou, L. Mamatas, S. Petridou, P. Papadimitriou, D. Moura, A. I. Swapna, B. Pinheiro *et al.*, "A marketplace-based approach to cloud network slice composition across multiple domains," in *IEEE Conference on Network Softwarization (NetSoft)*, 2019, pp. 480–488.

[19] R. Li, Z. Zhao, Q. Sun, I. Chih-Lin, C. Yang, X. Chen, M. Zhao, and H. Zhang, "Deep reinforcement learning for resource management in network slicing," *IEEE Access*, vol. 6, pp. 74 429–74 441, 2018.

[20] D. Dietrich, C. Papagianni, P. Papadimitriou, and J. S. Baras, "Network function placement on virtualized cellular cores," in *9th International Conference on Communication Systems and Networks (COMSNETS)*, 2017, pp. 259–266.

[21] C. Papagianni, P. Papadimitriou, and J. S. Baras, "Rethinking service chain embedding for cellular network slicing," in *IFIP Networking Conference (IFIP Networking) and Workshops*, 2018, pp. 1–9.

European Union
European Regional Development Fund

HELLENIC REPUBLIC
MINISTRY OF ECONOMY & DEVELOPMENT
SPECIAL SECRETARY FOR ERDF & CF
MANAGING AUTHORITY OF EPAnEK

EPAnEK 2014-2020
OPERATIONAL PROGRAMME
COMPETITIVENESS ENTREPRENEURSHIP INNOVATION

ΕΣΠΑ 2014-2020
ανάπτυξη - εργασία - αλληλεγγύη
Partnership Agreement 2014 - 2020

Co-financed by Greece and the European Union