Noname manuscript No.
(will be inserted by the editor)

Experimental validation of Resource Allocation in Transport Network Slicing using the ADRENALINE Testbed

Ricard Vilalta * · Raul Muñoz · Ramon Casellas · Ricardo Martínez · Fei Li · Pengcheng Tang

Received: 09.20.2019

Abstract Transport network virtualization provides the necessary data and control plane technologies as key enablers of future networks. The interaction between network slicing and optical-transport network virtualization architectures is under study to automate effective network resource orchestration.

In this paper, we present an harmonized network slicing and transport network virtualization architecture, including a network slice planner tool, which is designed and implemented enabling in-operation execution of network slice resource allocation algorithms. We validate the proposed architecture by providing a novel resource allocation algorithm, evaluating its performance, and deploying two different slices on top of the ADRENALINE Testbed, while measuring both slices key performance indicators.

Keywords NFV · Optical SDN

1 Introduction

A wide range of services and use cases that are being proposed from different vertical industries [1] will need to be supported by upcoming networks (both fixed and mobile). Each vertical industry service and use case imposes its own set of requirements to the underlying network infrastructure. These requirements can be described in terms of functional and non-functional requisites such as security, latency, elasticity, resiliency, bandwidth, etc. The same structural and functional network infrastructure must be able to fulfil these strin-

Ricard Vilalta *, Raul Muñoz, Ramon Casellas, and Ricardo Martínez
Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA)
Castelldefels, Barcelona, Spain
E-mail: ricard.vilalta@cttc.cat

Fei Li, and Pengcheng Tang
Huawei
Shangai, China

gent and varied requirements for transport networks. Transport networks and their supported services are being constructed upon two novel networking enablers: Software Defined Networking (SDN) and Network Function Virtualization (NFV). In this regard, SDN/NFV concepts complete the vision of clearly separating hardware and software. This separation allows service providers using appealing capabilities such as: i) network programmability throughout multi-vendor, multi-technology and multi-domain scenarios; and ii) virtualization of both functions and infrastructure to support modular, flexible and heterogeneous network services [2].

To deal with this heterogeneity of network services, the Next Generation Mobile Networks (NGMN) Alliance proposed the concept of network slicing [3]. A network slice instance is formed by a set of network functions, and the resources enabling the deployment of these functions, that form a complete instantiated logical network to meet certain network characteristics. The requested logical network instance allocated by the network slice may be managed/controlled by either the own user that has requested the network slice or delegated to the network provider(s) owning the resources supporting the slicing instance. To this end, the adoption of SDN and NFV solutions becomes essential to manage and configure those multiple logical infrastructures.

Network virtualization becomes a key enabler for network slicing since it provides the necessary technologies to program and secure underlying networks with the specified set of network requirements, while offering the necessary isolation for virtual links between network slices. In this paper, we will review the suggested technologies from both data and control plane perspectives.

Herein, we address the objective of dynamically computing, (re-)allocating and deploying network slicing instances (with specific requirements in terms of CPU, throughput, etc.) on top of cloud infrastructure, which is interconnected through a transport network that provides network virtualization. Cloud resources can be allocated in both intra- and/or inter-data centers. We have considered multiple NFV infrastructure, under a single administrative domain [4]. A slicing planner tool (referred to as NetSlice Planner) is designed to enable the in-operation execution of slice resource allocation algorithms. The input information to such a tool (and hence, the devised algorithm/s) is a detailed view of the availability of all the resources including both network and cloud. This paper presents a novel algorithm for resource allocation of requested network slices, which considers the most efficient use of resources in order to lower the network slice request blocking ratio. The output of this algorithm is formed by the selected logical resources (network and cloud) satisfying the network slice request.

This paper extends [5] by including an a novel transport network slicing resource allocation algorithm and its simulated evaluation and benchmarking, as well as an experimental validation of the proposed network slicing architecture on top of ADRENALINE testbed.

This paper is structured as follows. Section 2 provides more explicit details on the technologies for network virtualization. Section 3 details: a) architecture for network slicing, including a dynamic network slicing planner; b) a complete network slicing deployment workflow; and c) network slice resource allocation general problem, and in particular, it is described the devised network slicing resource allocation algorithm. Finally, Section 4 provides a novel experimental validation of the proposed network slice planner and novel results in key performance indicators in deployed slices in the ADRENALINE Testbed.

2 State of the Art

In this section, an state of the art on network virtualization techniques is firstly provided. Later, network slicing is introduced and several proposed architectures are described.

2.1 Network Virtualization

Network Virtualization (NV) is defined as the partitioning and aggregation of the physical infrastructure to create multiple co-existing and independent Virtual Networks (VN) on top of it. NV can be introduced at data plane with enabling technologies which support virtualization (at both packet or circuit based connections), or with resource virtualization at the control plane level [6]. The usage of such virtualization technologies in Network Slicing might accomplish benefits in terms of security, latency, elasticity, resiliency and bandwidth.

2.1.1 Path to Dynamic Programmable Transport Layer

At the data plane, NV can be performed differently according to the considered layer (Figure 1). Each proposed layer provides some degree of virtualization, thus providing multiple independent instances of the connectivity services that it supports, while providing isolation between these services.

At the Layer 0, dedicated physical interfaces might provide a per-port virtualization, thus assigning different ports to independent connectivity services. The same could be applied taking into consideration optical wavelengths, optical cores and optical modes which might be allocated to a dedicated VN.

At layer 1, OTN tunnels can be considered as independent connectivity services can be supported through them.

At the Layer 2, MPLS and Flex Ethernet connections can be adopted. Currently, MPLS-TP over DWDM is used to support the virtualization of the physical network resources to deploy per-tenant network tunnels over an optical infrastructure. In this context, Flexible Ethernet solution (FlexE) over OTN is an emergent evolutionary technology that is expected to be rapidly adopted. FlexE provides end-to-end connections, by providing a shim layer enabling the multiplexing in time of several Ethernet clients. The main advantage of FlexE is that each connection is served as a dedicated data path with deterministic (carrier-graded) performance. Deterministic latency and guaranteed bandwidth are provided to each connection for a tenant or service along with total data separation for privacy and security.

Moreover, at the layer 2.5, the use of VLANs allows also creating up to 4094 VNs over the same physical Ethernet interfaces.

Finally, at the Layer 3 and above, the composition of overlay networks through tunneling mechanisms (e.g., NVGRE [7], NSH [8]) provides the necessary network virtualization mechanisms that are being integrated.

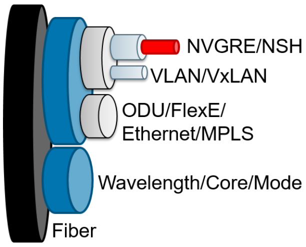

Fig. 1 Network Virtualization

2.1.2 Programmable Control Plane

From the control plane perspective, several initiatives are currently addressing the NV framework. A Virtual Transport Network Service (VTNS) is presented in [9] as the creation and offering of a VN by a provider to a user. VNs may be dynamically created, deleted, or modified and users can perform connection management, monitoring and protection within their allocated VNs. Different types of VTNS could be associated to operators offering, for example, Bandwidth on Demand (BoD) services, Network as a Service (NaaS) or Network Slicing for 5G Networking.

Multi-domain Network Hypervisor (MNH) [6] allows the direct handling of the allocated VN resources. MNH allows a tenant to independently control its allocated network resources through its own Customer SDN Controller (CSC). It interacts with a Multi-domain SDN Orchestrator (MSO) to provision the underlying end-to-end virtual links which eventually compose the targeted VN. Typically, the SDN controller of each VN runs in a dedicated host. It can be deployed using several available SDN controllers (e.g., OpenDaylight or ONOS).

In IETF, the Abstraction and Control of Traffic Engineered Networks (ACTN) architecture [10] defines the requirements, use cases, and an SDN-based architecture, relying on the concepts of network and service abstraction. The architecture encompasses Physical Network Controllers (PNCs) which are responsible for specific technology and/or administrative domains. PNCs are then orchestrated by a Multi-Domain Service Coordinator (MDSC). By doing so, MDSC enables abstraction of the underlying transport resources and deployment of VN instances for individual customers / applications, which are controlled by each individual Customer Network Controller.

Integration of current programmable control plane for transport network slicing, could be handled through the description of Virtual Link Descriptors (VLD), which might be generated through a WAN Infrastructure Managers (WIM) that is able to understand network intents in the form of ONF Transport API, or IETF ACTN data models.

2.2 3GPP Network Slices and ETSI NFV Network Services

3GPP has proposed a data model [11] for network slices. A network slice consists of a list of Network Slice Subnetworks instances (NSSI). Each NSSI contains a set of network functions and resources for these network functions being arranged and configured to form a logical network. Each Network Function (NF) can be either an access or a core network function. A network slice also includes all information relevant to the interconnections between those NFs like endpoint connectivity and individual link requirements (e.g., QoS attributes). A network slice instance is created by using a Network Slice Template (NST).

The data model of a network slice is mapped in [12] towards current ETSI NFV data models. It is actually highlighting the relationship between Network Services and Slices/Subnet_Slices. This is important since the NFV Orchestrator (NFV-O) is familiar and supports the Network Service (and VNF) constructions (and even NFVI-PoP interconnection). The virtualised resources for the slice subnet and their connectivity to physical resources can be represented by the nested Network Service concept, or one or more VNFs and Physical Network Functions (PNF) are directly attached to the Network Service used by the network slice subnet. ETSI states that "an NFV Network Service (NS) can thus be regarded as a resource-centric view of a network slice, for the cases where a Network Slice Instance (NSI) would contain at least one virtualised network function".

2.3 Proposed Network Slicing architectures and resource allocation algorithms

End-to-end network (E2E) slicing is described in [13] and many references to current SoA are provided in tutorial style. Several research projects built around the concepts of network slicing are presented, which focus on network slicing that includes both sliced RAN and multiple interconnected network services.

Several examples of E2E network slicing include [14], which analyzes the architecture of network slicing for micro and macrocells and it assesses the impact of increasing numbers of slices; or include inter-domain network slice deployment described in [15] and the architecture proposed in Multi-domain Network Slicing Orchestration Architecture [16].

Regarding resource allocation algorithms in network slicing, many papers focus on RAN and E2E network slicing, but few focus on transport network slicing, being one example [17], which presents network slice resource allocation and monitoring over multiple clouds and networks. Another paper presenting transport network slicing resource allocation architecture is [18].

In the reviewed literature, it can be observed that although resource allocation algorithms are significantly analysed for end-to-end network slices, the concept of a network slice planner is novel in architecture. Moreover, transport network slicing through the usage of a transport network for VIM interconnection is currently under study, which this paper provides novel results. Regarding resource allocation for network services, there is many literature available, but we have considered mainly [19] and [20].

3 Optical-Transport Network Slicing Resource Allocation

The proposed optical-transport slicing architecture aims at providing multiple, highly flexible, end-to-end network and cloud infrastructure slices operated in parallel over the same physical infrastructure to fulfil vertical-specific requirements as well as mobile broadband services. Moreover, the proposed architecture will also consider the relationship with WIM to provide the underlying network virtualization described in the previous section.

3.1 Network Slicing architecture

Figure 2 depicts the proposed network slice architecture enabling to: a) dynamically accommodate slicing requests; b) select resources (i.e., networking including required VN and cloud) to be allocated; and c) interact with the corresponding functions (i.e., SDN and NFV orchestrators) to actually convey the programmability and instantiation of the assigned resources.

To do so, the adopted architecture follows the slicing model presented in previous section. Basically, this approach provides multiple, highly flexible, end-to-end network and cloud infrastructure slices which are operated in parallel over the same (common) physical infrastructure to fulfil user (vertical-specific) requirements and services. The deployed architecture is formed by five key building blocks: a) NFVI-PoP, b) Virtualized Infrastructure Managers (VIMs) for the cloud resources; c) WIMs for network resources; d) NFV Orchestrator (NFVO); e) Slice Controller; and f) NetSlice Planner.

The NFVI-PoP is the computing, storage and networking infrastructure that is located in a single site and that it is offered for the deployment of VNFs and their interconnection. This networking infrastructure is typically based on a L2 network connectivity (e.g., VLAN) controlled through an SDN controller. The SDN controller uses L2/L3 network virtualization for the instantiation of

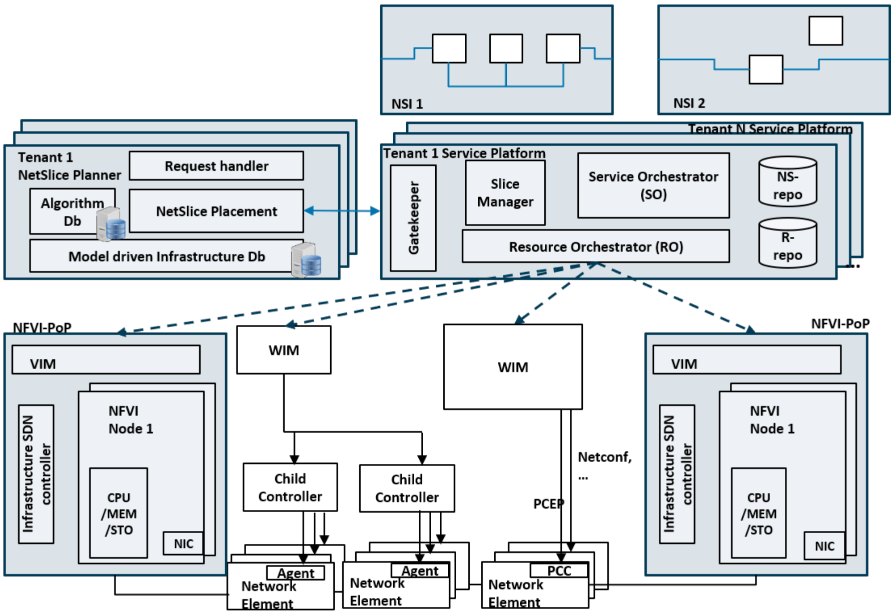

Fig. 2 Multi-VIM network slicing architecture including NetSlice Planner

different network services. Each NFVI-PoP offers its control and management interface towards the service platform through a VIM.

The VIM is able to handle multiple tenants in each NFVI-PoP. Well-known implementations of VIM are OpenStack and VMware. The VIM is responsible for the instantiation (i.e., the creation / deletion) of virtual machine instances hosting the required VNFs and network services for the requested slice service. The VIM also handles the storage of disk images, as well as managing the intra-data center network connectivity for each tenant. In this regard, if a number of VNFs are instantiated, the VIM configures the connectivity among the different functions following the required forwarding graph, which is known as service chaining.

The WIMs act as network controllers enabling the inter-data center connectivity and incorporating the benefits of network virtualization, as explained in Section 2. Specifically, slice's network functions can be allocated in a distributed way, that is, in different and geographically remote data centers (NFVI-PoPs). Thus, dedicated interconnection among such NFVI-PoPs is needed throughout a network infrastructure which is handled by the WIM(s). Typically, NFVI-PoPs are inter-connected on top of a software-defined metro/core network infrastructure which may combine multiple switching technologies such as packet and/or optical. Bearing this in mind, the WIMs are SDN controllers providing the necessary coordination, computation, selection and configuration (programmability) of the underlying network resources. Examples of WIM are ONOS [21] or OpenDayLight SDN controllers.

The Service Platform (that consist of NFV-O + VNF manager in ETSI NFV reference architecture) is the responsible for allocating the requested network slice instances and to deploy on top of them the necessary network services. It consists of: a) Gatekeeper, b) Service Orchestrator (SO), c) Resource Orchestrator (RO) and d) Slice Controller. The Gatekeeper module within the service platform processes the incoming and outgoing requests. The Slice Manager takes over of the mapping between the requested network slice and the NFV network services, while tackling network service lifecycle [12]. In other words, the Slice Manager plays the role of traditional NFV Operations Support Systems (OSS) requesting to the NFVO the instantiation of the whole network slice involving network service creation and the required network service inter-connectivity. The Slice Manager handles the slice life cycle interacting with the Service Orchestrator (SO). The SO receives the service packages and performs the placing, deploying, provisioning, scaling, and managing of the services within the existing cloud infrastructures. The Resource Orchestrator (RO) allows the service platform entities interacting with the infrastructure. It exposes interfaces to manage services and VNF instances, retrieve monitoring information about the infrastructure status, and reserve resources for services deployment.

Finally, the NetSlice Planner is a powerful tool that allows the computation and accommodation of dynamically arriving network slice requests. It consists on several components that enable this functionality, while providing the necessary tools for its own extension. Firstly, the algorithm database (Db) is the component that has the different programmed resource allocation algorithms for network slice allocation. The NetSlice placement uses an specific algorithm and it applies it upon a requested network slice (received by the request handler). Later, two algorithms are proposed and compared (i.e., First-Fit and FALCON). The model driven infrastructure database allocates information of the resource status of the different NFVI-PoPs, as well as the inter-connection network status (obtained through the different WIMs). Once an optimal resource allocation is computed, it is requested to the underlying Service Platform.

3.2 Proposed workflow

Figure 3 shows the workflow of the proposed architecture. It can be observed that in step A, the NetSlice Planner requests server information to the VIM (i.e., OpenStack) and network topology and resource state to the WIM (i.e., ONOS). This is the preliminary step, which is realized during the initialization phase. If more VIM and WIM are involved, the necessary information should be requested and retrieved.

The dynamic operation mode is triggered when a Network Slice is requested (Step B). A network slice, as mentioned, consists on several interconnected network services. Therefore, the NSD information is requested to the NFV-O to properly create the requested graph (Step C). When a request graph

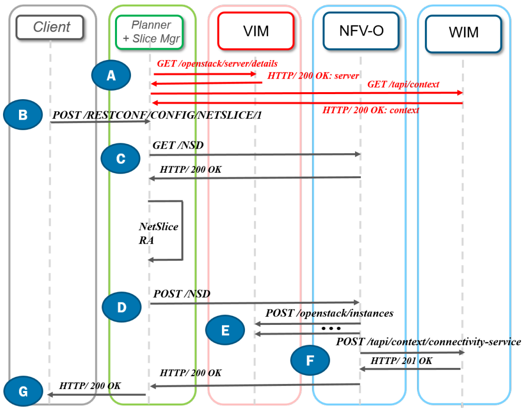

Fig. 3 Proposed Message exchange workflow

is obtained, the selected resource allocation (RA) algorithm is triggered to determine cloud and networking resources to be eventually allocated. The NFV-O is then commanded for creating the necessary network services (Step D).

The NFV-O handles the deployment of the necessary services on top of the selected VIM (Step E). Later, the necessary Virtual Link Descriptor (VLD) are requested through the WIM (Step F). Finally, Slice Manager acknowledges the received resource allocation and deploys it on top of the NFV-O (Step G).

3.3 Network Slice Resource Allocation

In this section we first describe the Network Slice Resource Allocation (RA) problem, as detailed in NetSlice Planner. To solve the RA problem, a heuristic algorithm based on a first-fit approach is proposed and used as a benchmark. Finally, the devised FALCON heuristic algorithm is presented. As shown below, FALCON leads to appreciably improve the performance for the Network Slicing RA problem attained by the first-fit approach.

3.3.1 Network Slice Resource Allocation Problem Definition

We identify the following data models:

- **Substrate infrastructure**. We model the substrate infrastructure as a directed graph and denote it by $G^S = (N^S, H^S, L^S)$, where N^S is the set of substrate switching nodes, H^S is the set of substrate hosting nodes (each compute node), and L^S denotes the set of substrate links $l^S = (u^S, v^S), l^S \in L^S$.
- **Network Slice request**. As previously explained, a network slice is considered as a directed graph of interconnected (shared) network services. In order to model a network slice, we denote by a directed graph $G^V =$

(H^V, L^V) the Network Slice request. H^V denotes the set of virtual hosts (e.g., virtual machines, VMs) and L^V denotes the set of virtual links between virtual hosts.

Now we define a set of capacity functions for the substrate and virtual resources. Each host (physical or virtual) $h^x \in H^x, x \in \{S, V\}$ is attributed with a set of A attributes (CPU resources, memory resources, and storage resources) whose capacities are denoted as $c_a(h^x), a \in A, h^x \in H^X, A = \{CPU, MEM, STO\}$ (we consider only CPU, memory and storage as host attributes). Each link $l^x \in L^X$ is associated with a bandwidth capacity $bw(l^x)$. Moreover, we define $av\_bw(l^x)$ as the available bandwidth capacity.

We also denote $P^S(l^V)$ as the paths in the substrate network that maps to the virtual link l^V.

The objective is to find a mapping function for all virtual hosts and links to the substrate infrastructure as:

$$M : (H^V, L^V) \mapsto (H^S, L^S)$$

3.3.2 First-Fit Network Slice Resource Allocation Algorithm

In order to provide a benchmark algorithm (First-Fit), we assess the problem in two steps:

- Step 1: Following a First-Fit procedure, we select the minimum number of substrate hosting nodes with enough capacity to allocate all the virtual hosts in H^V.
- Step 2: adopting a Shortest Path algorithm (e.g., Dijkstra) to find a feasible path in the substrate network for each l^V, considering the allocated substrate hosting nodes.

3.3.3 FALCON Network Slice Resource Allocation Algorithm

The conceived FALCON algorithm (described in Algorithm 1) evaluates each data center (DC) resources to allocate requests to those DCs where more resources are available. Because of not only DC resources are sufficient to serve the slice instance request, available network link capacity is also taken into account. In light of this, we propose an evaluation function that leverages both characteristics (see Equation 1).

$$F(DC, \alpha) = \sum_{\forall h \in H^S, DC} \alpha \frac{c_a(h)}{c(h)} + \sum_{\forall l \in L^S, DC} (1 - \alpha) \frac{av\_bw(l)}{bw(l)} \tag{1}$$

For each DC, the FALCON algorithm evaluates it. This evaluation allows the algorithm sorting the hosts. Once the hosts are properly ordered, the substrate hosts for the virtual hosts are allocated. Since the best substrate hosts have been selected, also considering link capacity constraints, next the shortest path algorithm is triggered for each of the requested virtual links.

Algorithm 1 FALCON(G^S, G^V)

```
Input: $G^S$: Substrate graph, $G^V$: Request graph.
Output: $M$: solution set
  for $dc$ in $DC$ do
      $F \leftarrow F(dc)$
  end for
  $H^V = Sort(F, H^V)$
  for $h^V$ in $H^V$ do
      $allocated\_node = False$
      for $h^S$ in $H^S$ do
          if $c_a(h^S) <= c_a(h^V), \forall a \in A$ then
              $h^S \mapsto h^V$
              $allocated\_node = True$
              break
          end if
      end for
      if $not\ allocated\_node$ then
          $error\_node\_allocation$
      end if
  end for
  for $l^V$ in $L^V$ do
      $(u^V, v^V) = (u^S, v^S)$
      $P = shortest\_path((u^S, v^S))$
      if $av\_bw(p) >= bw(l^V), \forall p \in P$ then
          $P \mapsto l^V$
      else
          $error\_link\_allocation$
      end if
  end for
  return $M : (H^V, L^V) \mapsto (H^S, L^S)$
```

The FALCON algorithm provides an in-depth analysis of the location of the resources in comparison with first-fit algorithm. This will result in lower network utilization and reduced latency, when evaluating algorithm results.

4 Experimental results

In this section first, we evaluate the performance of FALCON resource allocation algorithm using a simulation environment, and once assessed its benefits, we demonstrate the complete architecture including the NetSlice Planner running FALCON algorithm by experimental demonstration on top of the ADRENALINE Testbed.

4.1 Network Slice Resource Allocation

The adopted substrate infrastructure scenario for conducting the simulations is an extended version of the NSFNET formed by 14 nodes and 42 unidirectional links and 6 ditributed DCs as shown in Fig. 4.b. We have assumed a leaf-spine configuration for the DCs (see Fig. 4.c). Table 1 (left) details the configured

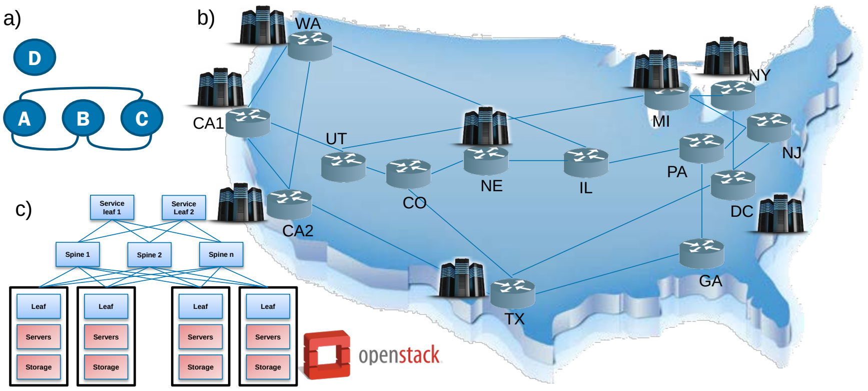

Fig. 4 a) Example of Network Slice Request; b) Simulated NSFNet topology including DC locations; c) Zoom-in of a DC base on leaf-spine topology

values for hosts per leaf, number of leafs and number of spines. The substrate infrastructure is initially configured with a predefined capacities which are maintained along all the performance data points. The values of the capacities of each DC are also uniformly distributed among the values included in each range depicted in 1 (left).

Parameter	Values
CPU per host	8
Memory per host	64
Storage per host	1000
Number of leaf	4
Number of spine	2
Hosts per leaf	48
ToR Bandwidth	10 Gbps
Leaf Bandwidth	100 Gbps
Spine Bandwidth	400 Gbps
Inter-DC Bandwidth	1000 Gbps

Parameter	Values
CPU	[1, 2, 4, 8]
Memory	[2, 4, 8, 16]
Storage	[20, 40, 80, 160]
Bandwidth	[0.1 - 1] Gbps
Nodes	[1 - 10]
Edge Probability	0.3

Table 1 Scenario parameter configuration (left) and Network Slice request parameter configuration (right)

In the Network Slice requests, the number of nodes is randomly determined by a uniform distribution between 1 and 10, as shown in Table 1 (right). Each pair of nodes are randomly connected with probability 0.3. The required capacities of both virtual hosts and links for each slice are as well selected randomly following a uniform distribution along the ranges depicted in Table 1 (right).

The inter-arrival process for generating the Network Slice requests is modelled by Poisson, whereas the duration of successfully deployed slice instance (holding time, HT) follows a negative exponential distribution. The average

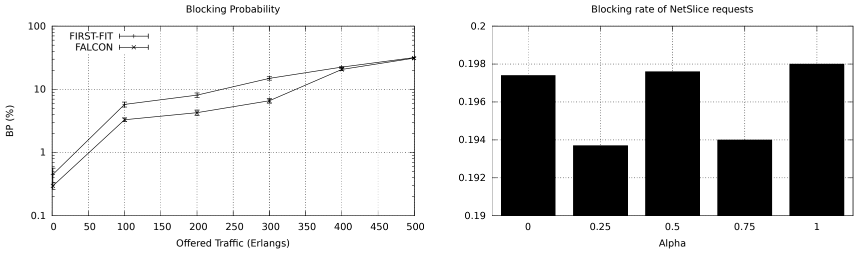

Fig. 5 Blocking rate of Network Slice requests (left) and Analysis of α in relation with blocking rate of Network Slice requests (right)

Inter-Arrival Time (IAT) is set to 1s, and the average Network Slice HT is varied for an offered traffic load ranging between 1 and 500 Er. 10^4 Network Slice requests have been generated for each obtained data point.

Figure 5 (left) depicts the blocking rate of the Network Slice requests in the NSFNET topology scenario, for the two Network Slice RA algorithms: First-Fit and FALCON (setting $\alpha = 0.5$). For a request load set to 1 Er, the attained blocking rate for the First-Fit and Falcon RA algorithm is 0.43%, and 0.27%, respectively. In general, for low to moderate slice request loads, we observe that FALCON algorithm is the most suitable. In particular, the most appreciable difference between both algorithms is obtained for a request load of 300 Er. In this load, the obtained blocking ratio are 14.51%, and 6.12% for First-Fit and FALCON, respectively. These rationale behind this performance difference can be explained because the FALCON algorithm leads to apply a more efficient use of the available resources (both compute and link) which favors the accommodation of subsequent incoming slice requests.

An analysis of the relationship of the α parameter with the blocking rate of Network Slice requests is provided in Fig. 5 (right). For a fixed load of 400 Er, several values of α have been considered to select the most appropriate value for the considered scenario. We can observe that for a value of 0.25, the blocking probability is minimal. This is given to the fact that the proposed scenario is constrained by the link capacity.

Figure 6 shows the blocking rate of Network Slice requests by reason for blocking in FALCON algorithm). It can be observed that under heavy load requests most of blocked Network Slice requests are because of Node-blocked requests.

To evaluate the algorithm complexity, we analyze the Network Slice RA time, understood as the necessary computation time to map a Network Slice request into substrate resources (i.e., compute and link). Figure 7 shows the RA time for both algorithms (First-Fit and FALCON). It can be observed that FALCON requires more RA time, as its complexity is increased, due to the analysis of the DC resources both in terms of compute and links.

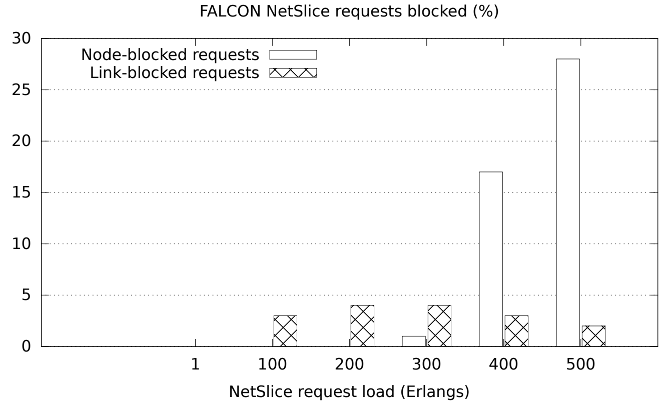

Fig. 6 Detailed FALCON blocking rate of Network Slice requests

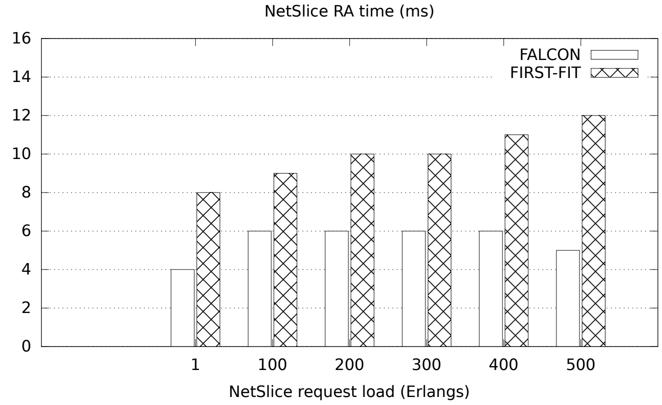

Fig. 7 Network Slice Resource allocation time (right)

4.2 Transport Control Plane Validation

In Section 3, we have presented the proposed architecture. In order to validate it, the cloud-computing platform and transport network of the CTTC ADRENALINE Testbed [22] has been used. Figure 8 shows the ADRENALINE testbed, which is composed of multiple components and prototypes, with the objective of offering end-to-end services. Moreover, users and applications are interconnected across multiple heterogeneous network and cloud technologies for the development and validation of network slices and services in conditions close to production systems.

ADRENALINE includes the following technologies/capabilities: a) A fixed/flexigrid DWDM core network interconnected with white box ROADM/OXC nodes; b) A packet-based transport network for the edge (access) and metro segments for traffic aggregation and switching of Ethernet and MPLS flows with QoS,

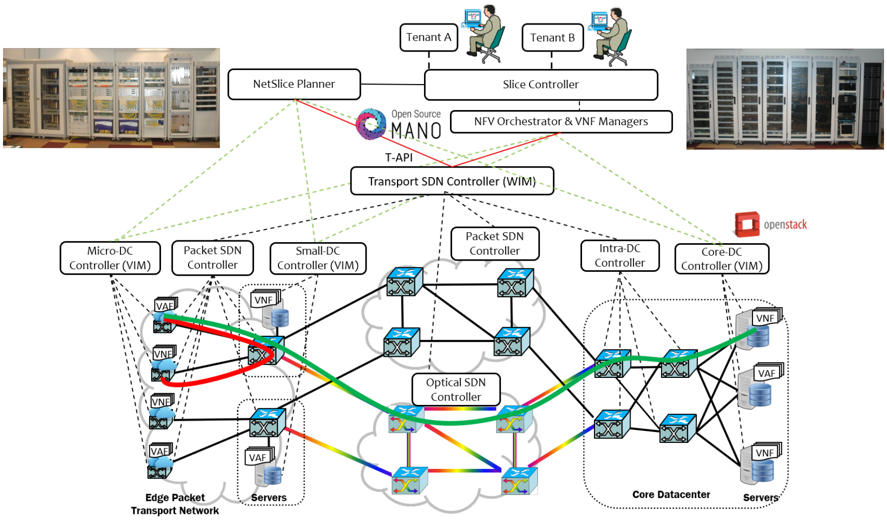

Fig. 8 ADRENALINE Testbed architecture and deployment of Slice A (in green) and Slice B (in red)

and alien wavelength transport to the optical core network; c) a distributed core and edge cloud platform; d) An SDN/NFV control and orchestration system for the joint orchestration of the multi-layer (packet/optical) network resources and the distributed cloud infrastructure resources; and e) Network Slices are offered as interconnected network services [23].

For our validation we have focused on the integration of the NetSlice Planner on top of the Slice Manager, which actually provides the dynamic life-cycle management (i.e., provisioning, modification and deletion) of network slices. Each network slice composed of virtual resources (network services form by a myriad of computing, storage and networking resources) exists in a parallel and isolated fashion for different tenants. Each tenant may represent vertical industries or virtual operators and has its own specific requirements (e.g, security, latency, resiliency, bandwidth). The NFV-O provides per-tenant programmability of the network slices and exposes an abstracted view of the network slices' virtual resources to each tenant. In our experimental validation, OpenSource MANO (OSM) [24] has been used as NFV-O and VNFM. OpenStack has been used as a VIM and NFVI-PoP.

In the proposed validation, we have defined two different network slices to be deployed on top of the ADRENALINE Testbed. Table 2 indicates the different properties of the requested slices. Slice A and B will consist of a single Network Service (NS) that consist on two VNFs and a VLD. Slice A will request a maximum latency of 200 ms, while Slice B will request a maximum latency of 5 ms.

Each network slice is requested to the NetSlice Planner, which follows the workflow described in Section 3.2. In this particular scenario, Figure 8 shows the deployed Slice A (in green). It can be observed that as the latency

	Slice A	Slice B
VNF	2	2
VLD	1	1
NS	1	1
Latency	200ms	5ms
Path	Electrical-optical -electrical	Edge nodes
NV tech.	MPLS & wavelength	MPLS
RTT (min, avg, max)	0.766/0.985/2.586 ms	0.153/0.209/4.914 ms
Jitter	0.063 ms	0.038 ms

Table 2 Slices properties and results

requirement was not stringent, a VNF has been allocated in the core data center (DC). The adopted network virtualization technology has been MPLS (layer 2) and a dedicated wavelength (layer 0).

Figure 8 shows the allocation of Slice B (in red), where all VNFs have been allocated in the edge of the network using MPLS as the network virtualization technology, which will satisfy the demand of attaining minimum latency.

Table 2 shows the obtained results of the deployment of two different slices in the ADRENALINE Testbed. Slice A has required a complex path consisting of electrical-optical-electrical switches. The optical path consists of more than 70 km. This has increased the measured round trip time (RTT) to and average of 0.985 ms. A shorter path has been allocated for Slice B, so that the introduced RTT has been of 0.209 ms.

5 Conclusion

We have presented and demonstrated network virtualization technologies as key enablers to support optical-transport network slicing. In this regard, we have discussed a general architecture for multi-site NFVI-PoP that supports NS and multi-tenancy has been presented. We have introduced the NetSlice Planner aiming at properly allocating in-operation Network Slice requests. After modelling the problem of transport network slice resource allocation, considering both intra-DC and inter-DC connectivity, we have presented two algorithms, that suit the proposed architecture to allocate network slice requests. These algorithms have been compared in terms of both blocking rate and execution time.

Finally, we have validated the proposed architecture on top of the ADRENALINE Testbed where it is demonstrated the feasibility of the proposed solution by deploying different transport network slices using the NetSlice Planner on top of our NFV/SDN testbed. KPI results are provided per slice.

As future steps, we consider necessary that algorithms for resource allocation optimization in network slicing should be implemented in the near future by the Service Orchestrators (e.g., SONATA NFV Service Platform, ETSI OpenSource MANO).

Acknowledgements The research leading to these results has received funding from EC H2020 5GPPP project 5GTANGO (761493) and MINECO project AURORAS (RTI2018-099178).

References

1. Alliance, N.: 5g white paper. Next generation mobile networks, white paper pp. 1–125 (2015)
2. Katti, R., Prince, S.: A survey on role of photonic technologies in 5g communication systems. Photonic Network Communications pp. 1–21 (2019)
3. Alliance, N.: Description of network slicing concept. NGMN 5G P **1** (2016)
4. Velasco, L., et al.: A control and management architecture supporting autonomic nfv services. Photonic Network Communications **37**(1), 24–37 (2019)
5. Vilalta, R., López-de Lerma, A.M., Muñoz, R., Martínez, R., Casellas, R.: Optical networks virtualization and slicing in the 5g era. In: 2018 Optical Fiber Communications Conference and Exposition (OFC), pp. 1–3. IEEE (2018)
6. Vilalta, R., Mayoral, A., Munoz, R., Casellas, R., Martínez, R.: Multitenant transport networks with sdn/nfv. Journal of Lightwave Technology **34**(6), 1509–1515 (2016)
7. Garg, P., Wang, Y.: Nvgre: Network virtualization using generic routing encapsulation. Tech. rep. (2015)
8. Quinn, P., Guichard, J.: Service function chaining: Creating a service plane via network service headers. Computer **47**(11), 38–44 (2014)
9. Virtual Transport Network Services Specification 1.0. Implementation agreement, OIF (2017)
10. Casellas, R., Vilalta, R., Martínez, R., Muñoz, R., Zheng, H., Lee, Y.: Experimental validation of the actn architecture for flexi-grid optical networks using active stateful hierarchical pces. In: Transparent Optical Networks (ICTON), 2017 19th International Conference on, pp. 1–4. IEEE (2017)
11. 3GPP: 3gpp tr 28.801 study on management and orchestration of network slicing for next generation network (2017)
12. ETSI: Report on Network Slicing Support with ETSI NFV Architecture Framework. Standard (2017)
13. Galis, A., Makhijani, K.: Network slicing landscape: A holistic architectural approach, orchestration and management with applicability in mobile and fixed networks and clouds. IEEE Network Softwarization (NetSoft) 2018 (2018)
14. Meneses, F., et al.: Micro and macro network slicing: An experimental assessment of the impact of increasing numbers of slices. Wireless Personal Communications **106**(1), 119–134 (2019)
15. Valcarenghi, L., et al.: A framework for orchestration and federation of 5g services in a multi-domain scenario. In: Proceedings of the Workshop on Experimentation and Measurements in 5G, pp. 19–24. ACM (2018)
16. Taleb, T., Afolabi, I., Samdanis, K., Yousaf, F.Z.: On multi-domain network slicing orchestration architecture and federated resource control. IEEE Network pp. 1–11 (2019). DOI 10.1109/MNET.2018.1800267
17. Martínez, R., et al.: Network slicing resource allocation and monitoring over multiple clouds and networks. In: 2018 Optical Fiber Communications Conference and Exposition (OFC), pp. 1–3. IEEE (2018)
18. Li, X., Casellas, R., Landi, G., de la Oliva, A., Costa-Perez, X., Garcia-Saavedra, A., Deiss, T., Cominardi, L., Vilalta, R.: 5g-crosshaul network slicing: Enabling multi-tenancy in mobile transport networks. IEEE Communications Magazine **55**(8), 128–137 (2017)
19. Xilouris, G., Kolometsos, S., Parada, C., Bonnet, J., Stavrianos, P., Kapassa, E., Touloupou, M., Kyriazis, D., Gouvas, P., Fotopoulou, E., et al.: Towards autonomic policy-based network service deployment with sla and monitoring. In: 2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 1–2. IEEE (2018)

20. Herrera, J.G., Botero, J.F.: Resource allocation in nfv: A comprehensive survey. IEEE Transactions on Network and Service Management **13**(3), 518–532 (2016)
21. Berde, P., et al.: Onos: towards an open, distributed sdn os. In: Proceedings of the third workshop on Hot topics in software defined networking, pp. 1–6. ACM (2014)
22. Muñoz, R., Nadal, L., Casellas, R., Moreolo, M.S., Vilalta, R., Fàbrega, J.M., Martínez, R., Mayoral, A., Vílchez, F.J.: The adrenaline testbed: An sdn/nfv packet/optical transport network and edge/core cloud platform for end-to-end 5g and iot services. In: Networks and Communications (EuCNC), 2017 European Conference on, pp. 1–5. IEEE (2017)
23. Mayoral, A., Vilalta, R., Casellas, R., Martinez, R., Munoz, R.: Multi-tenant 5g network slicing architecture with dynamic deployment of virtualized tenant management and orchestration (mano) instances. In: ECOC 2016; 42nd European Conference on Optical Communication; Proceedings of, pp. 1–3. VDE (2016)
24. ETSI, O.: Open source mano. OSM home page (2016)