Analysis of Network's QoS in Service Chains

Khalil Mebarkia, Zoltán Zsóka
Department of Networked Systems and Services,
Budapest University of Technology and Economics
{*mebarkia, zsoka*}*@hit.bme.hu*

***Abstract*—New network architectures like 5G adopt new service models where more providers share the infrastructure and offer different network services for users. The slicing concept in 5G networks is important to provide flexible, scalable, and on-demand solutions for the vast array of applications in networks. In this paper, we present a multilayer network model that allows the Service Function Chaining (SFC) with the support of multi-tenant slices. To analyze the Quality of Service in the slices, we set up a network with few virtual routers over Amazon Web Services (AWS) and use a variety of technologies such as Segment Routing (SR) for implementing Service Chains. We analyze SFC algorithms with different capabilities in avoiding overloads on the network links. The analysis shows the impact of traffic load increase on packet losses and the impact of deploying more slices on the same infrastructure.**

***Index Terms*—SFC, NFV, QoS, Slicing.**

I. INTRODUCTION

Evolved network architectures and technologies offer the more and more effective provision of a wide set of services. Complex use of the available networking, storage, processing functions, and resources allow the quality and performance required by the services. The changes of needs conclude in frequent reorganization and reconfiguration of the network functions and resources.

Network Function Virtualization (NFV) provides a way to flexibly use the resources and to realize the required Virtualized Network Functions (VNFs) in network nodes equipped with generic hardware. Service Function Chaining (SFC) concept and its deployments have seen amazing success in the context of NFV, for which the setup of services is typically deployed in virtual environments (e.g., virtual machines, containers). The services can be distributed and connected across a cloud infrastructure by a dedicated hypervisor or NFV orchestrator. Moreover, VNF-SFC builds a chain according to the VNFs required for the traffic and uses the network nodes where these functions are available, while orchestration includes the management of the devices that host the VNFs.

The concept of the 5G mobile networks includes slicing [1], i.e., the organization of network and computation resources into soft- or hard-separated sets according to services offered by the network. Different services might require different series of VNFs, and the network infrastructure and VNF resources might be or not be shared among them. SFC and orchestration for the traffic of a service have to be performed inside the assigned resource slice.

Many service providers and research institutes operate a single network infrastructure to support an ever-increasing number of services, thus the ability to fit transport customized to application needs is critically important. This includes creating network slices with different characteristics, which can coexist on top of the shared network infrastructure [2].

Thus, slicing cannot be limited only on the processing resources, SFC has to consider the transporting resources in the network, which can also influence the service quality. It implies the need for high flexibility in the configuration of network devices. An enabling technology is Software Defined Networking (SDN), but due to switching performance issues it is not supported in a large part of the transport and mobile backhaul networks [3].

A further important property of these networks is that the services and slices can be provided by different tenants, also referred to as virtual network operators (VNOs). On the other hand, the network infrastructure is not a huge integrated system. There are several infrastructure providers (InPs) that own and operate a subset of the resources.

As a consequence, the networking world is now driven by a completely different set of business needs, such as the ability to fully use network links, high performance, scalability, flexibility, high availability, simplicity in operation, and so on. This changed requirement needs automation of reconfiguration or managing the network by Software as with SDN.

Network Programmability has been steadily growing in the IT and networking communities. *YANG* can be used as a data modeling language suitable for automation and reconfiguration. For such purposes, *YANG* data models need to be accompanied with *NETCONF* and *RESTCONF* protocols [4]. Furthermore, these protocols enable the control and management of *YANG* data models, and also describe how to map a *YANG* specification to *RPC* or a *RESTful* interface that runs over *HTTP*.

Segment Routing (SR) is an architecture based on the source routing paradigm that seeks the right balance between distributed intelligence and centralized programmability by steering a packet through an ordered list of instructions. It is also an architecture that has been evolved from MPLS, a technology that overcomes the lack of flexibility and path control, and the performance issues of basic IPv4 routing solutions. The capability to use Traffic Engineering (TE) and resource reservation based on Resource Reservation Protocol with Traffic Engineering (RSVP-TE) lets us involve MPLS in the solution of SFC with slicing. In the context of SFC leverages on SR in such a way SF is associated with a segment that can then be used in a segment list to steer packets through

SummerSim-SPECTS 2020, July 20-22, Madrid, Spain; ©2020 Society for Modeling and Simulation (SCS) International

SF. Thus, SR provides a complete integration solution for SFC.

On the other hand, the drawback of MPLS is the higher cost of maintaining a more complex control plane. A solution for this issue can be the connection of all nodes in the network to one controller and make them all speak a common protocol, as realized for instance in Cisco devices. This way the tenant has to manage only the source router or the controller, which makes the infrastructure more programmable and scalable. Segment Routing applies this concept. In SR, the segments basically indicate the exact path sections that the packet has to follow, i.e., it considers a series of network nodes to touch. In other terms, segments are the instructions for the packet to follow to reach its destination.

The way of applying SR for slicing is illustrated on Fig. 1, which shows three different services realized in different slices.

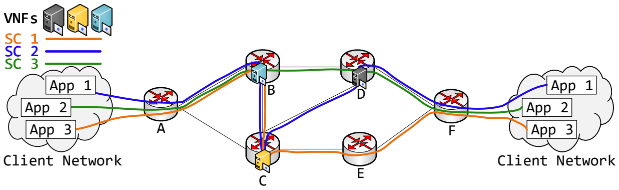

Fig. 1. Segment Routing in slicing

Motivated by all these contexts, this paper presents a model that is based on multilayer graphs and supports multi-tenant slicing in VNF-SFC. Our aim is to handle service demands from different slices using the same network infrastructure and execute SFC solutions that help to avoid network link overloads and QoS violations. Besides the model description, the solutions are analyzed on a simple but real network topology samples, where the service chains (SCs) are implemented using SR.

To achieve our aim, we set up our concept over Amazon Web Services (*AWS*) that provides virtual routers and servers which our testbed is built from. We use a variety of technologies that allow us to chose Segment Routing as a technique for realizing service chains in the network. Then, we test our multi-slice VNF-SFC concept by analyzing the network loads and packet losses in simple scenarios using static SFC and Service Traffic Engineering (STE) solutions. The name of the latter comes from the objective and concept similarities to the Traffic Engineering (TE) problem, and it is introduced in [5]. However, the support of different NFs and their given order makes the problem different from classic TE and thus, neither multilayer TE solutions are obvious to be applied for STE.

The evaluation of solutions proposed in this research area are mostly based on simulations and calculations, thus the use of the real implementation is a further contribution.

The rest of this paper is organized as follows. In Section.II, we present related works about the problem of multi-tenant slicing and its analysis. In Section.III, we define the problem and describe the proposed concept. We present the details of the implementation of our concept and of the performance evaluation method in Section.IV. Finally, we evaluate and analyze the results by presenting different measurements such as the throughput and packet loss in the Section.V.

II. RELATED WORKS

Papers [6], [7] address NFV as a promising architecture proposed to increase the scalability and the functionality of the network by leveraging virtualization technologies. It is the way in which telecommunication networks and services are designed and operated where traditional Network Functions (NFs) are transformed in VNFs such as Firewall (FW), Load Balancer (LB), or Network Address Translation (NAT) run over a distributed, cloud-based infrastructure referred to as Network Function Virtualization Infrastructure (NFVI).

Various works propose solutions for placement and chaining of service functions, which are the main problems in the SFC context. For instance the authors in [8] study the SFC embedding problem (SFC-EP) with dynamic VNF placement in geo-distributed cloud systems. They formulate the problem as a Binary Integer Programming (BIP) model aiming to embed SFC requests with the minimum embedding cost and proposed a novel method to efficiently embed SFC requests and optimize the number of the placed VNF instances. In [9], authors emphasize the importance of routing in the NFV network and propose a VNFs Routing Evaluation model. They aim to explore potentially better path by selecting those with minimum costs where they formulate the VNF routing problem as NP-hard aiming to minimize the routing cost.

Some recent papers address similar problems while considering multiple slices in the network. *Slice* can be defined as a set of network and VNF resources, which can support one or more services, each with a prescribed series of VNFs that the service traffic shall pass. The supported services can be told to *be in the slice*. The authors of [10] formulate the problem of statically embedding service chains into slices while considering also network link capacities. In [11] another MILP formulation is given for the problem of optimizing slices over multiple domains and accepting multiple services in each slice. The authors also present a heuristic that can guarantee the QoS requirements for the services by allocating the needed resources for the slices.

Many use cases have been proposed for 5G networking that promised to increase bandwidth and quality capabilities, and utilize the support for large numbers of end nodes. The authors in [12] investigate the service chain embedding problem for diversified 5G slice requirements, considering the sharing property of VNFs. They develop a fine-grained approach that considers resource requirements and limited traffic processing capacity of VNFs, which can be shared (or not) among slices depending on VNF functionalities.

Note that many works in the literature consider only one end-to-end VNF-SFC per slice, although the service demands belonging to a service supported by slice S^i may have different endpoint pairs. We have to assume that the VNF-SFCs of the demands in slice S^i can be different too.

In our previous work [13] we focus on 5G QoS issues like the end-to-end delay and loss which might come from the backhaul segment. We propose a multi-layer network model for the analysis of the effects of migrating the network to 5G, while considering different distribution and placement of VNFs in the IP network.

In [5], we address several challenges such as how to model the VNF-SFC problem considering the NF topology and current state of the network below it. It is also discussed, how to determine the SC according to the required bandwidth and VNF order while avoiding overloads on the network links. As a result, we propose a heuristic and ILP solutions to formulate these challenges. These solutions are based on the dynamic calculation of SC by considering the current network load to avoid the use of heavily loaded links. The heuristic algorithm *OdAASP* (Overload Avoiding Augmented Shortest Path) determines the shortest path between source and destination with the awareness of considering overload avoidance. As a comparative solution, we use the previously proposed algorithm *SFC-CSP* (SFC-Constrained Shortest Path) that finds the shortest path and satisfies a given SFC constraint.

A variety of works on Segment Routing implementations have been proposed and listed [14]. The authors of [15] present an SFC architecture based on SRv6 and an NFV infrastructure. They focus on the issue of steering traffic within a Linux-based NFV host that supports a potentially large number of VNFs. Note that, under particular Cloud environments such as *Amazon Web Services* (*AWS*), SRv6 cannot be implemented due to different kernel versions.

The authors in [16] study the SR optimization for VNF chaining with the objective to minimize the packet overhead of SR for all SFC demands. The problem is formulated as an Integer Linear Program (ILP), and a heuristic algorithm is proposed, which adopts the backtracking method to calculate a resource-efficient SFC path, and adopts the dynamic method to further compress the segment list.

III. SUPPORT OF SLICING IN VNF-SFC

A. Problem definition

We interpret the VNF-SFC problem as the search for a chain that leads the traffic of a service demand through network nodes with VNF capabilities. The chain shall go from the traffic source to the destination and touch VNFs in the order required for the service. It shall use the functional links connecting the VNF capable nodes, and due to the ordered series of VNFs there might be loops in it. There can be more than one nodes providing a given VNF.

To handle the QoS requirements of the service demands, the network infrastructure, and its relations to the functional links have to be considered. For this purpose the model presented in [5] includes a functional and a networking layer with a mapping of each functional link on a series of networking links.

The main challenge of modeling slicing in a network with multiple tenants and multiple InPs is to handle the resources of different companies. Obviously the referred multilayer model needs to be extended, since resources have to be assigned to distinguished slices that may belong to different organizations.

The problem of VNF-SFC with multi-tenant slicing support means a search for a chain similarly as above, but considering only the resources assigned to the tenant and slice that provides the required service. In our approach the functional links can be assigned to slices and no restriction is applied on the use of network resources.

Let us concentrate on the networking, and ignore the issues of reserving and operating the VNFs. The performance of the VNF capable nodes is considered to be unlimited, thus no quality issues come from those elements. On the other hand, also the rather complex problem of allowing multiple InPs is out of our scope for now.

Since more than one organizations play roles in the VNF-SFC with multi-tenant slicing, there are different schemes of information sharing among them. Neither the InPs nor other VNOs are obliged to inform a tenant about its own network structure and loads. From the level of information sharing between a VNO (functional layer) and an InP (networking layer) we distinguished three architectures in [5]: *overlay*, *integrated*, and *augmented*. From the point of view of the consideration of other slices and tenants in the VNF-SFC decision there are three applicable basic schemes:

1) The *independent* scheme does not allow any reservation, and load of other slices gets ignored too, i.e., during the VNF-SFC of a slice's demand only the functional resources of the slice are considered. This matches well in the overlay architecture, but can cause quality degradation due to overloads on networking links.
2) In the *static* scheme networking link bandwidth is divided and reserved for the tenants statically, and each tenant considers the reserved resources. The static assignment can lead to inefficient usage. On the other hand, the offline optimized bandwidth reservation is hardly applicable when user demands arrive in a dynamic manner.
3) In the *dynamic* scheme each tenant considers the current network load measured on links. This way the traffic load of the slices or even other tenants is taken into account in the VNF-SFC selection. This scheme requires an integrated cooperation between the physical and networking layers.

Many of the related works apply the independent or static schemes in slicing. In this paper, we adopt also the third scheme.

B. Solution concept

To embed multi-tenant slicing in the multilayer model for effective VNF-SFC we consider a slice as a set of VNF capable nodes, and links between these nodes, which can be optimally selected from the whole set of VNF-capable nodes and the connecting links.

Thus, our concept is rather clear: only a part of the whole functional graph is the functional graph assigned to a slice S_i. This part can be selected in advance according to the slice's resources. Obviously these resources do not belong to the InPs, but to the tenant of the slice, and their reservation can be dynamic, as described above.

Formally, for each slice S^i the functional layer $\mathcal{L}_F$ contains the graph $\mathcal{G}_F^i \subseteq \mathcal{G}_F$, and only $\mathcal{G}_F^i$ is considered when VNF-SFC is performed for a demand coming up in S^i. For each edge of $\mathcal{G}_F^i$, i.e., for each functional link, there must be a mapping on the edges of the graph $\mathcal{G}_N$ in the networking layer $\mathcal{L}_N$, i.e., on networking links.

There are two important cases from the tenant number point of view. If there is only one tenant:

- the functional graph $\mathcal{G}_F$ is sliced up into $\mathcal{G}_F^i$ subgraphs using the same set of nodes, but different sets of edges,
- VNF capabilities in node $v \in \mathcal{G}_F^i$ can be different for each i,
- the mapping of edge $f \in \mathcal{G}_F^i$ on edges of networking graph $\mathcal{G}_N$ is not dependent from i, i.e., two slices that use the same functional link in their SFCs use the same network resources,
- current capacity and load values of network link resources are available for all slices, when the used VNF-SFC is with the dynamic scheme,
- optionally the node capacities can also be shared, i.e., decisions should take into account the current capacities and loads of VNFs used by more slices, or VNFs can be assigned to individual slices.

In the other case, if there are multiple ($M > 1$) tenants which may not share VNF resources and support slices:

- an extension of the single-tenant case is possible by applying a functional graph $\mathcal{G}_{F,t}$ for each tenant $t \in 1..M$,
- the functional links belonging to different tenants can be mapped on either the same or on different networking links,
- the mapping scheme depends on the way that the InP uses in sharing its network resources among the tenants: it can be based on strict reservation, or on statistical multiplexing, applying queueing techniques with weighted service for the packets and optionally with priorities, e.g. WFQ or LLQ,
- the set of VNFs might differ for the tenants and their availability in the nodes can be also different,
- sharing of VNF resources among the tenants is not suggested,
- functional graph $\mathcal{G}_{F,t}$ can be then sliced into $\mathcal{G}_{F,t}^i$ subgraphs according to the slices of tenant t.

Fig. 2 illustrates our concept of using the layered graph with functional subgraphs for slices of a single tenant. [1] The edges of the networking graph are indicated with dashed lines, and those of the functional graph with solid and curved lines. For the sake of visibility, edges are not directed on the figure, although our model for VNF-SFC works with directed edges. In the mapping of each functional edge we apply here the shortest network path between its endpoints.

The example contains two slices: S_1 supports a service that requires VNFs a and b to pass, while the service in S_2 requires VNFs b and c to pass. The available VNFs are indicated near the nodes. On the figure we see how $\mathcal{G}_F$ is sliced up into $\mathcal{G}_F^1$ and $\mathcal{G}_F^2$ subgraphs that share the functional link between nodes v_5 and v_6. A rather important case is the use of edges v_1-v_5 in subgraphs $\mathcal{G}_F^1$ and v_3-v_5 in $\mathcal{G}_F^2$. These functional edges are dependent through the mapping on $\mathcal{G}_N$ since both will use the network link v_4-v_5.

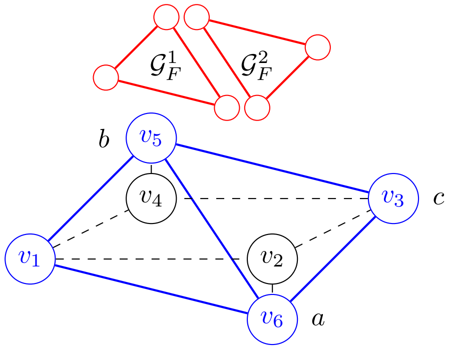

Fig. 2. The layered graph model of an example with two slices

IV. IMPLEMENTATION DETAILS

Our aim is to test our multi-slice VNF-SFC concept in a real network environment and analyze network loads and packet losses in simple scenarios using static and service traffic engineering SFC solutions. We have chosen the Segment Routing technique for realizing service chains in the network.

A. Concept Realization

Fig. 3 shows the building blocks of a realization of our multi-slice VNF-SFC concept. The most important block is the *Controlling* module, which calculates the VNF-SFCs. It applies the functional graph $\mathcal{G}_F$ considering its subgraphs $\mathcal{G}_F^i$ according to the slices, and the networking graph $\mathcal{G}_N$ with the current routing and load information.

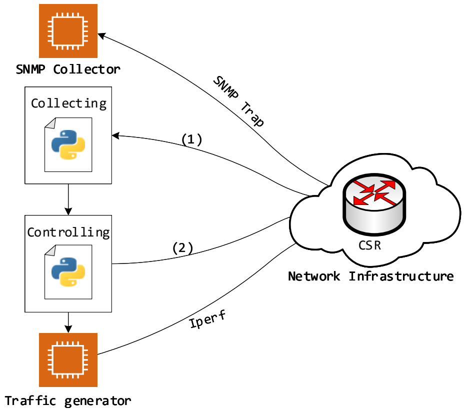

Fig. 3. Block diagram of the concept realization

Once the SC is found in $\mathcal{G}_F^i$, the next step is the creation of a *segment list*, in which the segments realize the whole series of the functional links in the SC, i.e., not only the nodes with

---

[1] Note that in this work we do not analyze the multi-tenant case.

applied VNF instances shall be listed. This is important, since otherwise the consideration of mapping on the networking layer cannot be ensured. The mapping of functional links can be realized through static or dynamic routing depending on the network settings of the InP.

The *Collecting* module is responsible for the querying of load and routing information. It performs *RESTCONF* queries on each router. In other words, *RESTCONF APIs* help to get the network's information such as the loads and capacities of links, and list of VNFs available in the *Head-end* and other routers (Step 1 in Fig.3). The information is proceeded to a separate server that runs a *Python* script with the *SFC* algorithm. The computed paths are posted to the appropriate routers (Step 2 in Fig.3).

B. Network topology

The above-listed modules are implemented in a network that is set up over the *AWS* cloud platform, which includes virtual routers and servers. The testbed is composed by 4 *CSR (Cisco Cloud Services Routers 1000V)* available on *AWS* marketplace with *IOS XE 16.10.b* software. The CSRs offer routing, security, and network management as cloud services with multitenancy which include the option of programmable configuration through RESTCONF.

The traffic endpoints are *Ubuntu 18.04 LTS Servers* where UDP traffic is generated using `iperf` that can produce standardized performance measurements for any network.

For collecting and organizing the information about routers on the IP network, we use Simple Network Management Protocol *(SNMPv2c)* in a separate server within *AWS*.

Since we focus on the loads and losses on networking links, we do not implement real VNF capabilities, neither in separated servers or attached datacenters, but assume the routers as VNF capable nodes. However, the Cisco Cloud Services Router 1000v (CSR 1000v) is a virtual-form-factor router that delivers comprehensive WAN gateway and network service functions into virtual and cloud environments. Note that if needed, a virtualized Linux-based environment can be used to implement VNFs even on a router, e.g. *Guestshell*, which is designed to run custom Linux applications on Cisco devices.

The implemented topology of the networking graph $\mathcal{G}_N$ can be seen in Fig. 4. There are four *Virtual Private Clouds (VPCs)*: (*VPC*$_1$, *VPC*$_2$, *VPC*$_3$ *and VPC*$_4$) which are connected respectively to each other via *VPC Peering*. This contains networking connections between *VPCs*, which enables to route the traffic using private IPv4 or IPv6 addresses within the *CIDR* range on *AWS*. Each *VPC* deploys an *EC2 instance* that hosts a single *CSR*. In the cloud environment the links of $\mathcal{G}_N$ can be realized as generic *GRE* tunnels between the routers.

The GRE tunnels are advertised by the *IS-IS* routing protocol to be used as parts of the *segments*, which are the realizations of the functional links in $\mathcal{G}_F^i$. Although the concept allows arbitrary mapping between the two layers, this time we use the simplest one-to-one mapping where each functional link matches one networking link (GRE tunnel). Thus, the calculated VNF-SFCs will be mapped directly on a series of networking links. As mentioned before, a segment route shall be configured for it, and it is implemented as an MPLS-TE tunnel and referred to as SR-LSP. It requires an *SR policy* configured on the *Head-end*. In this implementation, we use an *explicit path* for SR-LSP. Allowing segment routing solves the problem of leading the traffic through the network by touching the given series of routers (or other nodes), thus the configuration is needed only in the Head-end node. Note that this solution allows also network route loops in the SRs.

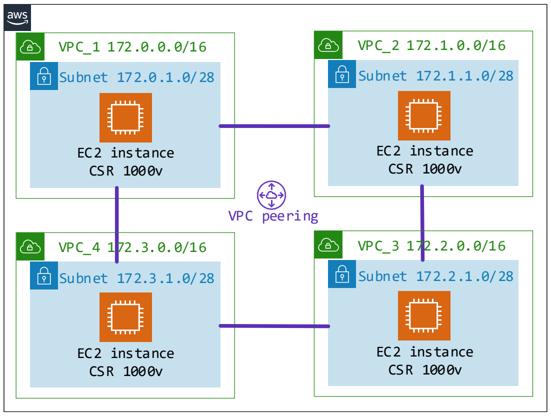

Fig. 4. Region and Availability Zone for the deployed instances

The GRE tunnels' capacities are configured and set to *10 Mbps*. However, these tunnels are virtual links, thus, even if the throughput reaches the maximum capacity, the tunnel might perform normally without any packet loss. To overcome this issue, we configure traffic policies that allow us to control the maximum rate of traffic sent or received on an interface by applying multiple classes of service.

In this work, we use Segment Routing IPv4 *(SR-IPv4)* implementation where *SR-MPLS* data plane must be enabled on all IPv4 interfaces in the *IS-IS* domain, since the available cloud environment suffers incompatibility issues with SRv6 of IPv6. This implies that a static route entry in the edge router shall drive the traffic with the given destination address to the right segment route.

Fig. 5 shows the network topology indicating only the networking links that will be loaded in our experiments. According to the mapping, segments (functional links) are created only between neighboring routers.

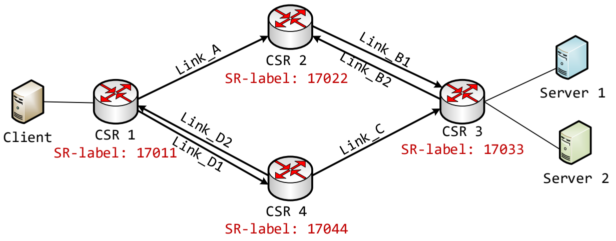

Fig. 5. Sample Network Topology

The traffic of a user demand instance in slice S_i starting in PC_{src} in subnet LAN_{src} and ending in PC_{dst} in subnet LAN_{dst} takes the following path in the networking layer:

1) In the (edge) router R_{src}, which is connected to LAN_{src} the traffic gets classified and assigned to the route configured via an MPLS-TE tunnel. The classification is not connected to a reservation process and is restricted to take decisions only on the base of the destination IP address.
2) The dynamic routing is applied only for the networking links of $\mathcal{G}_N$, which connect the routers, i.e., the GRE tunnels. The subnets of the PCs are not advertised. The route for the SR-LSP is built from these taking the link costs into account. The SR-LSP ends in the (edge) router R_{dst}, which is connected to LAN_{dst}.
3) In router R_{dst} the tunneled traffic has to be routed to the destination. In this sample case the subnet of the destination PC_{dst} is directly connected to the edge router.

Note that, in provider networks the last step can be performed with use of Virtual Route Forwarding (VRF) which is a technique widely applied for creating virtual private networks (VPN) over the same infrastructure. However, slicing is not obliged to VPNs, and in the general cases there are no private networks that might use overlapping address ranges and need to be separated. Instead of the complex configuration of VRFs, we can use static routes to forward the packets to their destinations.

C. Evaluation Methods

In this subsection, we present the different experiment setups used for evaluating the performance of the applied SFC algorithm. In both scenarios, our aim is to analyze the effect of serving dynamically arriving semi-permanent (never closing) service demands. This behavior is modeled with the generation of the traffic of the amount that is stepwise elevating in time.

Fig.6 introduces *Scenario 1* with one slice (Slice0) and one service. The SC requires to pass VNFs V_1, V_2 and V_3 (grey, yellow and blue).

According to the VNF requirements of the service there are two possible SCs implemented as *SR-LSPs* based on the proposed algorithms in [5]. On the one hand, the algorithm *SFC-CSP* consist to find the shortest path that satisfies a given SFC constraint. While the algorithm *OdAASP* finds another shortest path to a destination by taking into account the overload avoidance on the links. The *SR-LSPs* is organized as the following:

- *SR-LSP*$_1$ corresponds to the shortest path based *SFC-CSP* algorithm *(CSR*$_1$*, CSR*$_2$*, CSR*$_3$*, CSR*$_2$ *and CSR*$_3$*)*.
- *SR-LSP*$_2$ corresponds to the *OdAASP* algorithm that tries to avoid overload network links in SFC *(CSR*$_1$*, CSR*$_4$*, CSR*$_1$*, CSR*$_2$ *and CSR*$_3$*)*. The path is calculated in a multilayered graph which is built up based on VNF requirements and network's state. The algorithm considers the mapping of functional links and excludes the paths that might cause overloads on network links.

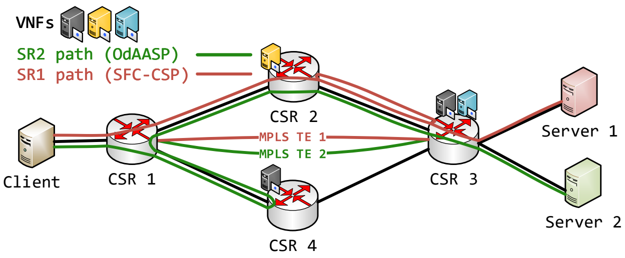

Fig. 6. SCs in Scenario 1

In this scenario, we want to mimic that when *OdAASP* cannot avoid overloads on links due to high traffic load, the SFC for a new traffic demand returns to be the shortest path chain (*SFC-CSP*). Our aim is to not overwrite the *SR-LSP* of the already flowing traffic demands. As shown in Fig. 7 we organize the SC choice by generating two different types of traffic and elevating the traffic load with the following phases:

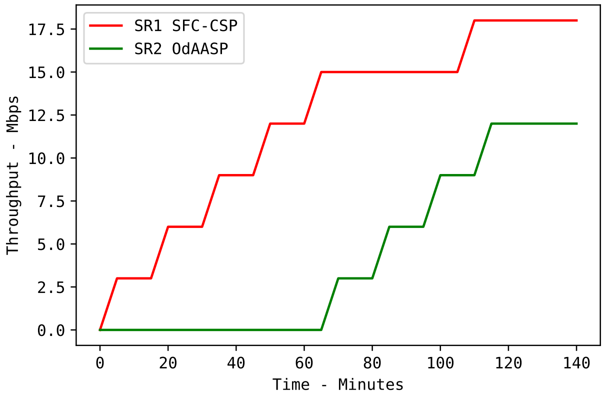

Fig. 7. Traffic generation in Scenario 1

- **Phase.I (SFC-CSP growing)** consists of generating a linearly growing UDP traffic load that takes the SR-LSP$_1$ path. This phase finishes when at least one of the network links would get overloaded. It is expected that the link $B1$ gets overloaded sooner than others as the traffic passes twice on it.
- **Phase.II (SFC-CSP constant, OdAASP growing)** consists, on the one hand, of generating a linearly growing UDP traffic load that takes the SR-LSP$_2$ path. On the other hand, modeling the unchanged SC of earlier arrived demands, the client keeps generating a fixed amount of traffic, which is being sent on path SR-LSP$_1$. This way we avoid the overload of link $B1$ for a while. The phase finishes when at least one of the network links would get overloaded.
- **Phase.III (SFC-CSP growing, OdAASP constant)** models the return to the shortest path SR-LSP$_1$ when the overload cannot be avoided.

Fig.8 introduces *Scenario 2* that extends the previous case of VNF-SFC to two slices (Slice0 and Slice1). As shown on Fig.8 the second slice supports one service using the SR-LSP

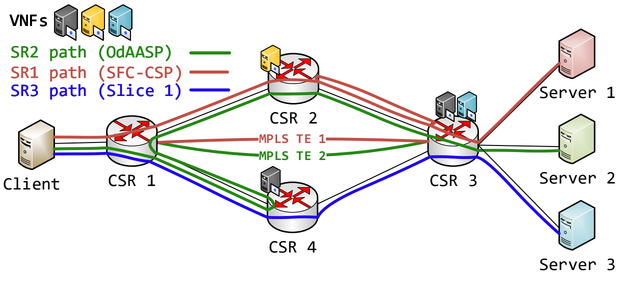

Fig. 8. SCs in Scenario 2

path *SR-LSP*$_3$. The SC in this second slice requires to pass VNFs V_1 and V_3 (grey and blue).

As it can be seen in Fig.9, the generation of traffic from the second slice follows a simpler way. It takes three different levels in the three phases determined as in Scenario 1. Note that the starting points of the phases here can differ from those implied in Scenario 1, since the traffic of the two slices might overload the network links earlier.

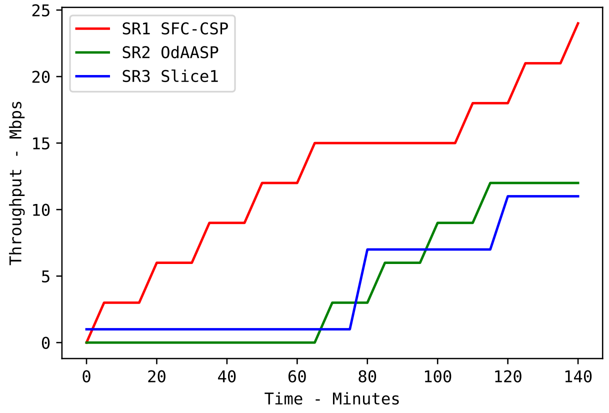

Fig. 9. Traffic generation in Scenario 2

V. MEASUREMENT RESULTS

The aim of using the *OdAASP* algorithm is to avoid network link overloads and this way get a better QoS, e.g. suffering less packet losses on links. Based on the statistics collected by the Collecting module we analyze the load and packet loss on network links indicated also on Fig. 5.

Fig.10 and 11 present the values measured with Scenario 1.

From $(0{-}25\mathtt{min})$ we observe that the load is increasing on links A, B_1, B_2 and C. Particularly on Link B_1, the increase is faster, because the SR-LSP$_1$ path passes link B_1 twice. Therefore, in Fig.11 we start observing packet loses on link B_1 after $25\ \mathtt{min}$ as a consequence of reaching the overload point at the link.

It is obvious to see the throughput on links A and B_1 have reached *10 Mbps* (Fig.10) alongside we observe packet loss within the same period of time because the the links' (Fig.11), because of the tunnels' virtual links as it is mentioned in Section.IV-B.

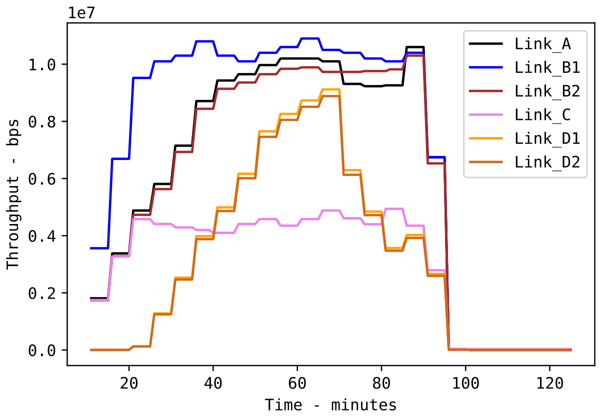

Fig. 10. Scenario 1: network link load

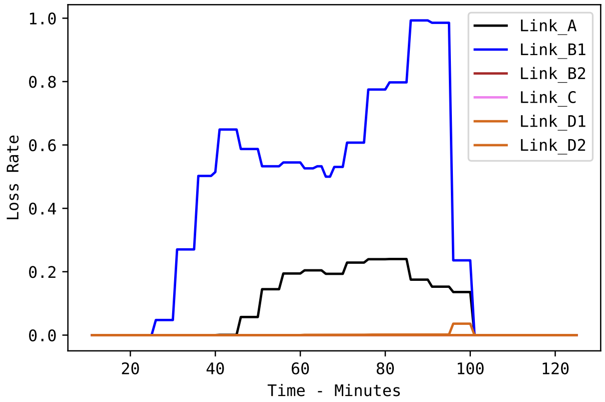

Fig. 11. Scenario 1: network link packet loss

Within the following period of time $(25{-}45\mathtt{min})$, the load increases on links D_1 and D_2 as the *Head-end* router imposes a new *MPLS* label stack that corresponds to SR-LSP$_2$, which involves the links D_1 and D_2. Furthermore, we observe that on link C the increase of load stops. Another interesting effect is the moderation of load and packet losses on link B_1, which comes from the moderation of traffic arriving twice to the link, because of the packet drops on this link.

At $45\mathtt{min}$, the Link A gets overloaded, and as consequence packets are being dropped.

On the one hand, within the following period of time $(45{-}90\mathtt{min})$ we observe that the links A, B_1 and B_2 are fully loaded and cannot bring more traffic, thus packet drops are unacceptably high.

Although, the measurements with using only the shortest path based SR-LSP$_2$ SC are not shown. It is easy to see that increase of load and packet loss on links A, B_1, B_2, and C would be linear for the whole measured period. This results in a much worse QoS than that experienced when involving SR-LSP$_1$, too.

In the evaluation of Scenario 2 we only focus on the changes that happened in the links (throughput and packet loss). We find similarities to the results of Scenario 1, because the load of some links are not at all affected when adding the *SR3* path. The differences are explained and shown in Fig.12 and Fig.13.

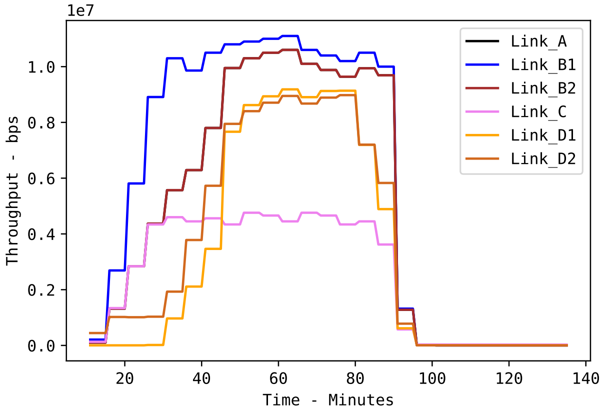

Fig. 12. Scenario 2: network link load

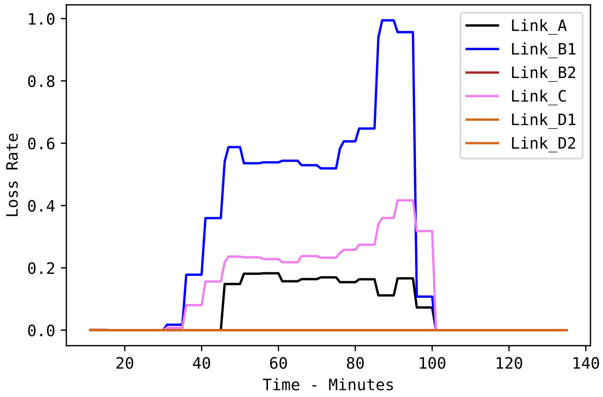

Fig. 13. Scenario 2: network link packet loss

In Fig.12, similarly to the previous experiment from `(0−30min)` we observe that the load is increasing linearly on the links A, B_1, B_2, C and D_1). Obviously, the Link C and D_1 are having a traffic passing through because the SR-LSP$_3$ path imposes the *MPLS* label of links. However, the load on link D_1 is slightly a fixed amount according to the of the traffic generation scheme of the second slice.

At `(30min)` in Fig.13, the links A and B_1 start getting overloaded and consequently packets are being dropped.

Within the period of time `(30−45min)` we observe that the load on links A and B_1 is increasing. As a consequence, the link B_1 is dropping packets with an increasing rate.

In the next period of time `(45−90min)`, we see a rather high, but stable value of load and loss.

We observe that the throughput on both Fig.10 and Fig.12 has exceeded the link B_1 capacity within particular intervals because the link itself consumed all the allocated capacity. Therefore, we observe packet loss within the same intervals in Fig.11 and Fig.13 respectively.

We conclude after these both experiments that the proposed solutions and concept are performing better to solve the avoidance load as long as the network load is moderate. Also, the concept based on these experiments seem to be performing even better in multi-slice situations.

VI. SUMMARY

In this paper, we have proven a concept of a small network realization applying Segment Routing. We have used a Service Function Chaining algorithms that try to avoid overloads on the network links. We have shown that segment routing architecture has helped to determine the path for the packet to follow from source to destination in really implemented Service Chaining scenarios. We have presented two different slices and observed the impact of slices on each other. The comparative results have shown the importance of the proposed solutions to avoid overloads in IP links.

REFERENCES

[1] D. Sattar and A. Matrawy, “Optimal slice allocation in 5g core networks,” *IEEE Networking Letters*, vol. 1, pp. 48–51, June 2019.

[2] Z. Ali, P. Camarillo, C.Filsfils, and D. Voyer, “Building blocks for slicing in segment routing network,” July 2, 2018.

[3] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, “5g network slicing using sdn and nfv: A survey of taxonomy, architectures and future challenges,” *Computer Networks*, vol. 167, p. 106984, 2020.

[4] K. W. K. W. A. Bierman, M. Bjorklund, “Restconf protocol,” *Internet Engineering Task Force (IETF)*, January, 2017.

[5] K. Mebarkia and Z. Zsoka, “Service traffic engineering: Avoiding link overloads in service chains,” *Journal of Communications and Networks*, vol. 21, pp. 69–80, Feb 2019.

[6] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, “Network function virtualization: Challenges and opportunities for innovations,” *IEEE Communications Magazine*, vol. 53, pp. 90–97, Feb 2015.

[7] R. Mijumbi, J. Serrat, J. 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, pp. 236–262, Firstquarter 2016.

[8] H. Tang, D. Zhou, and D. Chen, “Dynamic network function instance scaling based on traffic forecasting and vnf placement in operator data centers,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, pp. 530–543, March 2019.

[9] S. Zhang, Y. Liu, X. Gao, J. Zheng, and G. Chen, “Provably efficient algorithms for vnf routing optimization,” in *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 152–159, Dec 2018.

[10] T. Truong-Huu, P. Murali mohan, and M. Gurusamy, “Service chain embedding for diversified 5g slices with virtual network function sharing,” *IEEE Communications Letters*, vol. 23, pp. 826–829, 02 2019.

[11] R. Addad, M. Bagaa, T. Taleb, D. L. Cadette Dutra, and H. Flinck, “Optimization model for cross-domain network slices in 5g networks,” *IEEE Transactions on Mobile Computing*, pp. 1–1, 2019.

[12] T. Truong-Huu, P. Murali Mohan, and M. Gurusamy, “Service chain embedding for diversified 5g slices with virtual network function sharing,” *IEEE Communications Letters*, vol. 23, pp. 826–829, May 2019.

[13] K. Mebarkia and Z. Zsoka, “Qos modeling and analysis in 5g backhaul networks,” in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 1–6, Sep. 2018.

[14] Z. N. Abdullah, I. Ahmad, and I. Hussain, “Segment routing in software defined networks: A survey,” *IEEE Communications Surveys Tutorials*, vol. 21, pp. 464–486, Firstquarter 2019.

[15] A. Abdelsalam, F. Clad, C. Filsfils, S. Salsano, G. Siracusano, and L. Veltri, “Implementation of virtual network function chaining through segment routing in a linux-based nfv infrastructure,” in *2017 IEEE Conference on Network Softwarization (NetSoft)*, pp. 1–5, July 2017.

[16] Y. Wang, X. Zhang, L. Fan, S. Yu, and R. Lin, “Segment routing optimization for vnf chaining,” in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pp. 1–7, May 2019.