Traffic Steering for Service Function Chaining

Hajar Hantouti, Nabil Benamar, Tarik Taleb, and Abdelquoddous Laghrissi

***Abstract*—Dynamic service function chaining (SFC) is a technique that facilitates the enforcement of advanced services and differentiated traffic forwarding policies. It dynamically steers the traffic through an ordered list of service functions. Enabling SFC capabilities in the context of a software defined networking (SDN) architecture is promising, as it takes advantage of the SDN flexibility and automation abilities to structure service chains and improve the delivery time. However, the delivery time depends also on the traffic steering techniques used by an SFC solution. This paper provides a closer look at the current SDN architectures for SFC and provides an analysis of traffic steering techniques used by the current SDN-based SFC approaches. This study presents a comprehensive analysis of these approaches using efficiency criteria. It concludes that the studied solutions are not efficient enough to be deployed in real-life networks, principally due to scalability and flexibility limitations. Accordingly, this paper identifies relevant research challenges.**

***Index Terms*—Service function chaining, network services, traffic steering, NFV and SDN.**

I. Introduction

THE LARGE-SCALE networking infrastructures, provisioned by network operators and service providers, are typically designed to support a broad range of complex services and middle-boxes (MBox). MBoxes serve for performance and security enhancement as well as for the conformity of diverse policies. As stated by Sherry *et al.* [1], the number of MBoxes approximates the number of switches. With such a high number, the manual configuration and management of middle-boxes is tedious and error-prone. It becomes highly more complex in the case of large-scale networks.

Service Function Chaining (SFC) is a set of operations to direct traffic through an ordered list of MBoxes (or Service Functions – SF). SFC assists service providers and network operators in optimizing resource usage as a function of the number and nature of services to be delivered [2]. The service functions that compose a SFC are mainly located in an MBox (e.g., Firewall, Deep Packet Inspection - DPI, and Network Address Translator - NAT).

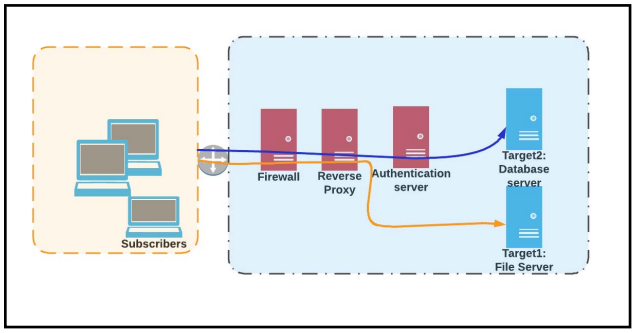

Fig. 1. Static service function chaining, placing service functions, one next to the other, to form a service function chain.

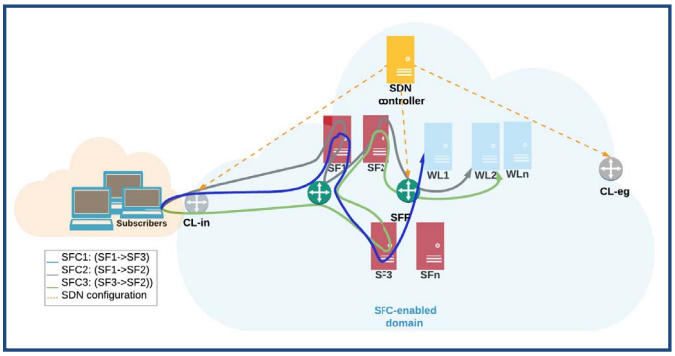

Fig. 2. Dynamic service chaining; an example of service function chains in an SFC-enabled domain. The SFs do not have to be located next to each other. Traffic is dynamically steered to the desired SFs.

Current SFC techniques often rely on static configuration tasks, such as configuring VLANs to steer the traffic through the desired set of SFs (as depicted in Fig. 1). Thus, the manual configuration and management of SFCs introduce a considerable complexity to the network operators.

With the emergence of Software Defined Networking (SDN) [3] and Network Function Virtualization (NFV) [4], the ability to dynamically structure SFCs and accordingly derive traffic forwarding policies becomes a reality (see Fig. 2).

On the one hand, SDN is a networking solution, characterized by the separation of the control plane from the data plane. Such separation facilitates the network programmability using an SDN computation logic that is centralized

Manuscript received September 22, 2017; revised March 3, 2018 and June 4, 2018; accepted July 8, 2018. Date of publication August 7, 2018; date of current version February 22, 2019. This work was supported in part by the Academy of Finland Project CSN under Grant 311654, and in part by the European Union's Horizon 2020 Research and Innovation Program through the 5G!Pagoda Project under Grant 723172. *(Corresponding author: Hajar Hantouti.)*

H. Hantouti is with the Department of Mathematics and Computer Science, School of Sciences, Moulay Ismail University, Meknes 11201, Morocco (e-mail: h.hantouti@edu.umi.ac.ma).

N. Benamar is with the Department of Computer Engineering, at the School of Technology, Moulay Ismail University, Meknes 11201, Morocco (e-mail: n.benamar@est.umi.ac.ma).

T. Taleb is with the Department of Communications and Networking, School of Electrical Engineering, Aalto University, 02150 Espoo, Finland, and also with the Computer and Information Security Department, Sejong University, 143-747 Seoul, South Korea (e-mail: tarik.taleb@aalto.fi).

A. Laghrissi is with the Department of Communications and Networking, School of Electrical Engineering, Aalto University, 02150 Espoo, Finland (e-mail: abdelquoddouss.laghrissi@aalto.fi).

Digital Object Identifier 10.1109/COMST.2018.2862404

and fed by various input data, such as network-originated notifications [5]–[7]. Programmable networking devices, such as SDN-enabled switches, allow programming SFC rules by an intelligent SFC engine that runs in the control plane [8].

On the other hand, NFV is a technique that aims to virtualize networking services to run on commodity hardware. NFV promotes running virtual appliances of SFs and creating virtual links between SFs which enhances SFC deployment. SFC is further simplified by using an NFV orchestrator to allocate resources, place SFs, and create virtual links [9]. Thus, NFV and SDN techniques can be used to facilitate the SFC design and operations.

SFC is a hot topic and tackles several problems [10] including: SFC path selection [11]–[13], service chain decomposition [14]–[17], SFC routing logic [18]–[20], optimization of SF placement in an SFC-enabled domain [12], [21]–[29], service allocation and provisioning [24]–[30], policy enforcement [31], troubleshooting [32], and security [33]–[38]. In spite of that, there are very few surveys about SFC. Recently, Bhamar *et al.* [39] presented a survey on SFC, discussing optimization approaches and outlining SFC challenges and open research issues. Although they mentioned the dynamic traffic steering, no thorough discussion about this topic was provided. Lately, Xie *et al.* [40] presented a survey about resource allocation algorithms for SFC. They reviewed a number of approaches: however, the survey focuses on resource allocation. Medhat *et al.* [41] presented a state of the art analysis on SFC from a performance and architecture viewpoint. This paper specifically focuses on traffic steering for SFC. Given the fact that the performance and efficiency of any SFC solution rely on the effectiveness of the underlying traffic steering techniques, and given the lack of consideration for the subject, it is clear that traffic steering issues merit further investigation.

The contributions of this paper can be summarized as follows:

- Description of the traffic steering operations and classification of different traffic steering methods into three categories: (i) based on packet headers, (ii) based on packet tags, and (iii) based on personalized SDN switching functionalities.
- Comprehensive analysis of a list of SDN-based SFC approaches, focusing on the traffic steering method used and presenting the advantages and the limits of each one. The approaches are classified based on the aforementioned traffic steering types.
- Qualitative evaluation of SDN-based SFC approaches, investigating the efficiency of the studied approaches. The evaluation is based on four criteria: deployment cost, flexibility, scalability, and overhead. Each criterion is composed of a set of metrics.
- An analysis of the evaluation setup used by the studied SFC approaches.
- An outline of ongoing research challenges and a proposal for future research directions.

To the best knowledge of the authors, this survey is the first to focus on SFC from the viewpoint of traffic steering. In particular, it analyzes the impact of traffic steering mechanisms on the efficiency of service chaining. The list of abbreviations used throughout this paper is presented in Table I.

TABLE I
LIST OF ABBREVIATIONS

Acronym	Description
CAPEX	Capital Expenditures
CL	Classifier
DPI	Deep Packet Inspection
DS	Differentiated Services
ETSI	European Telecommunication Standards Institute
GPE	Generic Protocol Extension for VxLAN
GRE	Generic Routing Encapsulation
IETF	Internet Engineering Task Force
IP	Internet Protocol
LISP	Locator/ID Separation Protocol
LSP	Label Switched Path
MAC	Media Access Control
MBox	Middle Box
MEC	Multi-access Edge Computing
MPLS	Multiprotocol Label Switching
NAT	Network Address Translation
NETCONF	Network Configuration Protocol
NFV	Network Function Virtualization
NFVI	Network Function Virtualization Infrastructure
NFV-MANO	NFV Management and Orchestration
NGSON	Next-generation Service Overlay Network
NSH	Network Service Header
OPEX	Operational Expenses
PCE	Path Computational Element
SC	Service Chain
SCH	Service Chain Header
SC-ID	Service Chain Identifier
SD-MBox	Software Defined Middle Box
SDN	Software Defined Networking
SF	Service Function
SFC	Service Function Chaining
SFCEH	Service Function Chaining Extension Header
SFF	Service Function Forwarder
SFP	Service Function Path
SPRING	Source Packet Routing in Networking
SR	Source Routing
SRE	Source Routing Encapsulation
SRH	Segment Routing Header
TCAM	Ternary Content-addressable Memory
TOS	Type Of Service
TS	Traffic Steering
VLAN	Virtual Local Area Network
VNF	Virtual Network Function
VXLAN	Virtual Extensible Local Area Network

The rest of this paper is organized as follows. Section II presents an overview of the concept of SFC, the related technologies and the motivation of research in SFC. Section III outlines the traffic steering types and methods used in SFC approaches. Section IV describes different SFC approaches in the context of SDN as well as the traffic steering methods used. Section V presents a qualitative assessment of the current SDN-based SFC approaches, using performance and

efficiency metrics, and discusses the validation techniques used by the studied SFC approaches. In Section VI, we discuss the results and present insights on the ongoing research challenges. Finally, the paper concludes in Section VII.

II. Service Function Chaining and Related Technologies

A. Service Function Chaining Concept

SFC is a networking concept that refers to the traversal of network traffic through a set of network services (i.e., any network service from the OSI layer 2 to 7) or service functions. Mainly for security reasons, a network operator may deploy firewalls and proxies, stitched together in the edges of the network, to prevent attacks. This process is known as the static SFC whereby networking services are placed one next to the other to provide a sophisticated service (see Fig. 1). Today, large-scale data-centers and Internet Service Providers (ISP), among others, express the need for a dynamic operation to achieve SFC, to reduce configuration and management complexities (see Fig. 2). Dynamic SFC is a research area that is developed with the emergence of new networking technologies such as cloud computing, SDN, and NFV.

B. SFC Architecture

RFC 7665 [2] defines SFC as a three layers architecture; a data plane, an SFC overlay, and a control plane. The first is the network layer composed of networking devices and the corresponding interconnections. The second is the layer consisting of SFC elements that are involved in the SFC operations, such as traffic classifiers (CL), service function forwarders (SFF), and SFs. The third contains the policy decision points managing the SFC overlay and data plane.

The purpose of SFC overlay is to guarantee an independence from the underlying physical network, thus, avoiding static configurations to achieve the dynamic service function chaining. Such independence permits network elasticity and management flexibility.

The SFC operation is initiated by the policy decision points that prepare the Service Function Paths (SFP) and profiles for a traffic classier. The latter filters the traffic and identifies the flows and the corresponding service chains. The identified flows are forwarded to an SFF that routes traffic flows to the corresponding next SFs (see Fig. 3). Usually, the connection between SFC elements is built based on network tunneling techniques such as VxLAN, GRE, and SFC encapsulation protocols (e.g., Network Service Header - NSH [42]).

C. Software Defined Networking

SDN relies on the separation of control and data planes. This separation enables network programmability, which allows to dynamically allocate resources and enforce policies based on the nature of required services [3]. The separation allows intelligence to be moved from devices to a control plane that manages the overall devices. The communication between the control plane and the forwarding plane is ensured by a southbound communication protocol, such as the OpenFlow protocol (OF) [43] and the Network Configuration protocol (NetConf) [44].

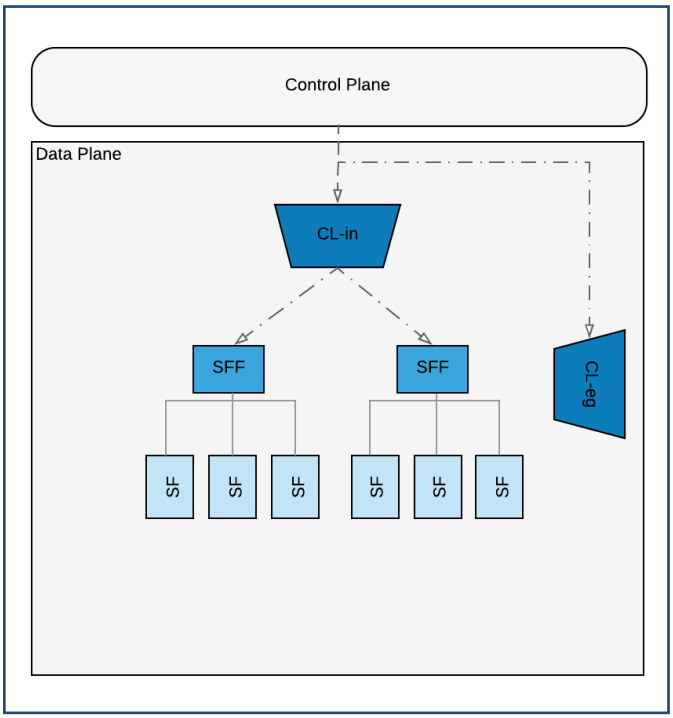

Fig. 3. Service Function chaining architecture as in RFC 7665 [2].

The commonality of SDN and SFC is that both are based on policy constructs. There is an upper layer above the control plane, which is composed of business applications, referred to as the management layer. The management layer sends applications requests to the control plane. The latter translates the policy requirements into classification rules (see Fig. 4). For example, some services require Internet-originated traffic to be inspected by a DPI instance. Thus, a service chain can be dynamically structured by the computation logic of SDN to include a DPI SF that will be invoked for each packet of such traffic. Deploying the service chaining above an SDN architecture permits to create and control a service chain, using software, in abstraction from the underlying topology.

D. Network Function Virtualization

NFV is a framework standardized by the NFV Industry Specification Group [45], chartered recently by the European Telecommunication Standards Institute (ETSI) [46]. NFV aims at reducing the cost of large-scale networking infrastructures, by virtualizing SFs that are usually supported by technology-specific devices. Virtualization is also supposed to facilitate a more agile deployment of resources. The contribution of NFV to the cost reduction needs more assessment [4]. The NFV framework consists of three core components: Virtual Network Functions (VNF), Network Function Virtualization Infrastructure (NFVI), and management and orchestration architectural framework (NFV-MANO). VNFs are software appliances of network functions, to be deployed in an NFVI. The NFVI consists of the environment that VNFs are built in, including hardware and software systems. This environment is managed and orchestrated by the NFV-MANO [4], [47].

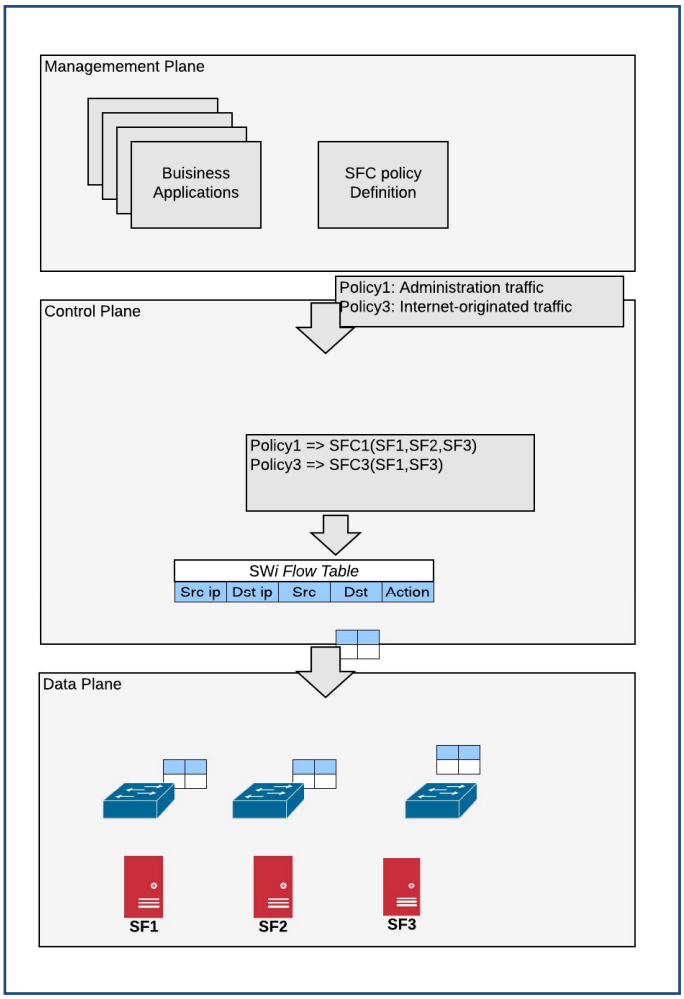

Fig. 4. Dynamic service function chaining in the context of SDN.

In modern networks, SDN and NFV support the deployment of SFC [48], [49]. The flexibility of SDN for SFC is enhanced with NFV through the possibility of dynamically instantiating and managing the virtualized service functions [50]. Furthermore, deploying SFC in an NFV environment permits to reduce the deployment cost by reducing the hardware investment. NFV also permits enhancing the system flexibility by managing service functions and resources in an on-demand manner [51].

III. Traffic Steering for SFC

The traffic steering for SFC refers to the operations involved in directing the traffic to reach the intermediate service functions that are involved in a specific service function chain. It can be defined as the forwarding and routing logic of traffic among SFs. Traffic steering inherits several functionalities from traffic engineering protocols and can be enhanced with SDN. The traffic steering functionalities for SFC are usually located in SFFs, and/or located in CLs.

As there are several traffic steering techniques, this section reviews a list of traffic steering methods of interest for SFC. A comprehensive list of traffic steering methods is categorized into three types: the first is based on packet headers for traffic steering, the second is based on specific tags or re-interpreting some packet fields, and the third is based on personalized switching functionalities and daemons (see Fig. 5). Later, the described types will be used to classify the SDN-based SFC approaches described in Section IV.

A. The Traffic Steering Operation

The process of forwarding or steering traffic for SFC includes the following operations: classification, identification, routing, and sometimes encapsulation.

The classification process permits filtering different traffic types according to policy profiles. To avoid re-classification at every SFC element, an identification process takes place for the next SFC element to process packets based on the result of the classification process. Usually, chain/path identifiers are inserted in packets or installed in SFFs [2].

The routing logic for SFC is a complex operation where two levels of routing take place: (i) an overlay routing between SFC elements connected through tunnels (i.e., when encapsulation mechanisms are used) and (ii) an underlay routing to ensure network reachability (e.g., L3 IP routing, Label Switching Path routing). The SFC routing operation is based on flow identifiers of traffic types or by matching rules in SFFs. The routing operations result in selecting the next SFC element in the SFP. In case a path identifier and/or metadata are added to the packets, an encapsulation process takes place to ensure connection and delivery between SFC elements [2].

We distinguish between two types of encapsulation: transport encapsulation and SFC encapsulation. The network encapsulation creates tunnels between different SFC elements and it can be any network transport protocol, such as IPinIP [52], VXLAN[53], VXLAN-GPE [54] or GRE [55], while SFC encapsulation refers to the information added to a packet to identify the SFP, such as NSH [42]. Encapsulation is used to achieve SFC while inserting path information within the packets. The use of transport and/or SFC encapsulation is approach dependent, as there are approaches that do not use encapsulation.

B. Header-Based Methods

SFC aims to achieve customized forwarding based on different policies. The policies are translated as chains of required SFs. Information about the forwarding path can be shared via additional packet headers. SFC headers are used between SFC entities such as CLs, SFFs and SFs to exchange SFC information and determine how to forward the different traffic types.

The early contributions for SFC and traffic steering started within the IETF Service Function Chaining [56] and ETSI [46] working groups. Several propositions are based on network headers such as NSH [42], Service Chain Header (SCH) [57], IPv6 extension header for SFC (SFCEH) [58], Segmented Routing Header (SRH) [49], or by defining a new IP option field.

1) Network Service Header: NSH [42] is an SFC forwarding protocol. The main functionality of NSH is to ensure traffic traversal to the required SFs. NSH protocol refers to a network

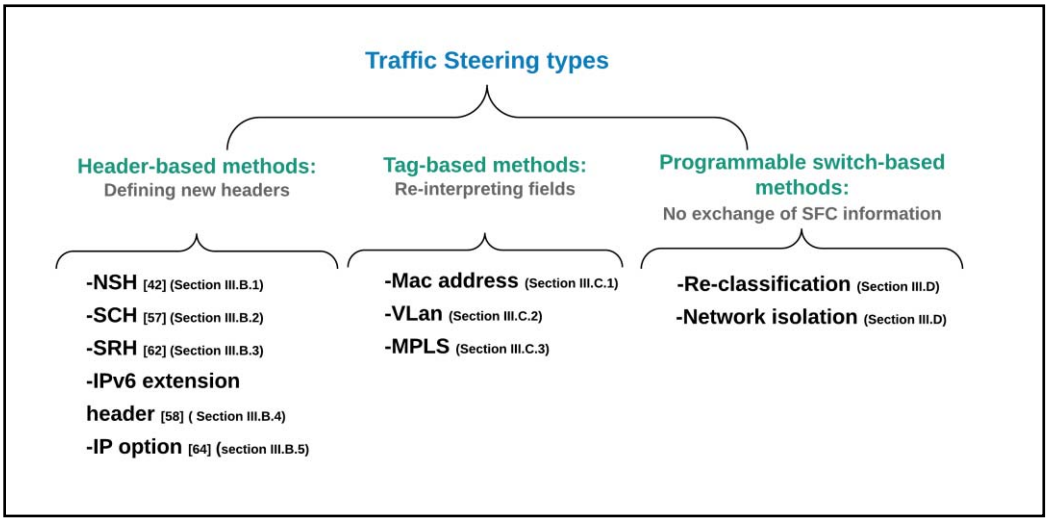

Fig. 5. Classification of traffic steering types.

header that is inserted into the traffic packets to ensure SFC elements traversal and SFC traffic steering. A CL classifies traffic and adds NSH headers that are used by SFFs and SFs to identify the traffic paths and share metadata. It should be noted that for legacy nodes, such as routers and switches, which may not be NSH aware, the use of network encapsulation or tunneling protocols becomes a necessity. NSH can be encapsulated in different tunnels, such as VxLAN and GPE. NSH was used as the traffic steering protocol in various SFC solutions, such as the work of Li *et al.* [36] and Mehmeri *et al.* [59]. Differently, Kulkarni *et al.* present in [60] a modification on the header to reduce the complexity and the cost of this protocol. By replacing the Service Path Identifier field in the NSH header with a Service Chain Identifier, they present a scalability enhancement using the modified header.

2) Service Chain Header: Similarly, SCH [57] is another header-based scheme relying on the concept of packet headers used in service chaining between different SFC elements. SCH is an alternative to NSH, yet adopting a different approach. SCH contains two types of fields, a mandatory and an optional field. As for the mandatory fields, it encodes the flow identifiers that permit for SFFs and SFs to steer traffic. The optional fields are useful to encode differentiated metadata.

3) IPv6 Extension Header for SFC: Another proposition to use packet headers for SFC traffic steering is an IPv6 extension header [58] that enables SFC in IPv6 enabled networks. It is based on a similar concept to NSH, and SCH in encoding path information (i.e., SFP Identifier, SFC identifier, and SF identifier), and optional information for metadata shared among the SFC elements. The extension header proposition requires a network encapsulation mechanism to make abstraction of the underlying network, and achieve connectivity between SFC elements to form the SFC overlay. Such encapsulation mechanisms include IPv6-IPv6 or IPv4-IPv6 tunnels [61].

4) Segment Routing Header: SRH [62] is employed by Abdelsalam *et al.* [63] as an SFC protocol. The IPv6 extension header for segment routing [62] is a routing protocol that can be used to achieve SFC. The authors proposed to encode the SFP in SRH. The IPv6 addresses of the involved SFC elements are written in the header. Thus, the CL inserts the corresponding header, and the SFFs steer the traffic based on the IP addresses encoded in the header [63].

5) IP Option Field: A new IP option field is proposed by Li *et al.* [64], to encode a list of SF and path identifiers corresponding to a specific traffic class. When packets reach an SFF, it checks the identifier and the connected SFs. This operation is repeated until the traffic reaches the final destination.

6) Advantages and Disadvantages of Header-Based Methods:

Advantages: Sharing SFC information in a dedicated header permits to safely exchange forwarding information. Usually, only SFC entities support the header and can process and update it.

Disadvantages: The major shortcoming of defining new headers is the need for SFs to support the header. Using proxies with each SF is inflexible and can increase deployment complexity and communication overhead. Increased packet size and encapsulation schemes in such methods can induce communication overhead as well.

7) Comparison Between SFC Headers: We notice that the headers NSH [42] and SCH [57] have a similar format. The IPv6EH [58] for SFC differs by encoding the SC identifier instead of the SFP identifier. These three headers include optional fields to encode further metadata. The SR header for SFC [62], [63] encodes the list of IPv6 addresses of the VNFs in the header and operates in a different way than the previous three headers (see Section IV-A). The size of the added packet header depends mainly on the optional metadata added, yet the SRH can have more size compared to the other headers if it encodes multiple IPv6 addresses. All the described headers require support from SFFs and SF or the use of proxies between SFs and SFFs.

C. Tag-Based Methods

Another method to determine an SFC path is to use existing packet fields. Packet fields can be re-interpreted to carry SFC

forwarding information. The information carried in these fields is called a tag. In this category of traffic steering methods, available packet fields in different layers are used to contain SFC information, such as MAC addresses, VLAN IDs, and MPLS labels.

1) MAC Address-Based Approaches: Metadata, containing SFC information on how to reach the next service function or to identify the service chain, can be encoded using MAC addresses. The approaches presented in [65], in addition to the approach defined in [66], are based on MAC addresses.

2) VLAN Based Methods: Several SFC approaches are based on VLAN IDs to encode tags [67]–[71], whether as local identifiers that the SFF uses locally to identify flows inbound and outbound of the SFs or SFFs, or as a user identifier [72]. The use of the tag is specific to the traffic steering designed by the SFC solution.

3) MPLS Based Method: Another way to encode SFC information is using MPLS Labels. It is either carried out by the use of MPLS in traffic engineering through Labeled Switched Paths (LSP), or using MPLS coupled with other tags to increase the SFC header capacity to encode more SFC related information [71].

Traffic engineering techniques can be customized for SFC purposes, such as the source routing technique discussed in the SPRING working group [73]. An SFC proposal in SPRING networks based on MPLS is described in [74] for SFC traffic steering. This technique relies on the source routing applied to MPLS. It proposes a LSP that encodes the list of SFs and SFFs of an SFP as a stack of MPLS labels. The traffic steering operation is initiated by a CL that encodes the SFP as a list of segment identifiers corresponding to the necessary SFs and SFFs to steer a packet through a service chain. It is worth noting that each SFF strips its identifier. The last SFF in the path is used to strip the last label in the LSP and forwards the packet to the final end-point. This protocol also relies on network encapsulation to forward LSP packets that can be encapsulated in MPLS-in-IP or MPLS-in-GRE tunnels. Thus, the underlying network can be an MPLS as it can be an IP network.

4) Advantages and Disadvantages of Tag-Based Methods:

Advantages: Tag-based identification of service chains enables the use of available packet fields for SFC purposes. Thus, it prevents the complexity of defining and supporting new headers.

Disadvantages: The ability to re-interpret available packet fields comes with a shortcoming; such fields can be used for other policy purposes. For example, using TOS field for SFC purposes might interfere with QoS policies deployed in the network. Thus, values defined for QoS can be interpreted as SFC values and vice versa, which can lead to misleading policies. Moreover, tagged packet fields might not be supported by all SFs, e.g., Some SFs might not support MPLS. Moreover, some SFs modify packet fields, which leads to loss of SFC information and the traffic will be steered differently.

D. Programmable Switch-Based Methods

The third type of traffic steering methods relies on programmable switches, mainly with multiple flow tables. Programmable switches play the role of SFC CLs and SFFs. Traffic forwarding is usually based on re-classifying traffic based on programmed flow rules, such as access lists. Network isolation can be used as well. For example, the work in [75] proposed the use of a switch and a controller per user to simplify user traffic identification. The identification of SFs can be based on port numbers or interfaces or based on the re-classification of packets (e.g., 5 tuple). Traffic steering can be based on coupling the previous methods, relying on tags or packet headers, with programmable switches.

Advantages: Determining the SFC path based on programmable switches can only be safe in a way that no manipulation of packets is performed. Switches are responsible for determining the direction of traffic. This method prevents switches from modifying each traversed packet. Thus, the cost of queuing and processing, resulting from packet modifications, is saved.

Disadvantages: Reclassifying flows traversing every switch is costly. It requires installing an important number of flow rules in different switches to match packets traversing each switch and determine the service chain and/or the SFC path. The increased number of rules influences the switches memory, processing, and overall switches performance [76], thus increasing the packet delivery time.

E. Summary and Insights

In this section, various traffic steering types were described. They were classified in three categories: the header-based traffic steering, the tag-based traffic steering, and the programmable switches-based traffic steering. This classification is to simplify the discussion, though the traffic steering can be hybrid. For example, an approach can rely on re-interpreting packet-field and a programmable switch, to rewrite the fields and further identify traffic destination based on switch ports, and an SDN controller to generate the SFC information [65]. Furthermore, two SFC headers can be implemented and used for different reasons, as in [63] where the SRH is used for sharing the forwarding path along with the NSH for sharing the context information. Moreover, multiple tags or fields can be combined in a traffic steering method to encode the SFC information [71]. Table II presents a summary of the different traffic steering types, with the relevant works, limitations, and strong points. The choice of the traffic steering method should be based on the infrastructure policy and context, as the limits of a traffic steering method can be accepted in a given context and not in another one, and the benefits of a method can be relevant for an infrastructure and not for another.

IV. SDN-Based SFC Approaches

Several proposals to dynamically chain services have been proposed in the last few years by the open source community, industry, and academia. In this paper, we mainly focus on academic research propositions to realize SFC. In this section, we present a comprehensive list of SDN-based SFC approaches and discuss the traffic steering methods used. The approaches are classified according to the traffic steering types described in Section III. Table III summarizes this classification.

TABLE II
SUMMARY OF TRAFFIC STEERING TYPES

Traffic steering type	Definition	Relevant works	Noticed limitations	Strong points
New headers	Defining new headers to share the SFC information between SFC-enabled devices.	[42][57][62][63][58][64]	-Devices need to support the headers. -Double encapsulation. -Increased packet size.	- Avoiding reclassification. -Sharing Metadata and context information.
Tags or packet fields	Re-interpreting a field in the packet header or adding tags to encode the SFC forwarding information.	[65][66][67][68][69][70][71] [72]	- Interference between policies using the tags for different reasons. -Some tags may not be supported by SFs.	-Avoiding reclassification. -Avoiding network encapsulation.
Programmable switches	Forwarding is based on specific configurations sent to the switches by an SDN controller.	[75][82]	-Reclassification. -Potential number of forwarding rules.	-Avoiding network encapsulation.

TABLE III
SDN-BASED APPROACHES AND RELATED TRAFFIC STEERING METHODS

SDN based SFC approach	Type of Traffic steering technique	SFC Metadata type
Pawar et al. [70]	Based on tagged packets	VLAN ID
Martini et al. [69]		VLAN ID
Abujoda et al. [71]		VLAN ID
Trajkovska *et al.* [72]		VLAN ID
Qazi et al. [67]		Vlan ID/MPLS label/PV4 TOS
Fayazbakhsh et al. [68]		IPv4 TOS/VLAN ID/IPv6 Flow Label
Blendin et al. [65]		Mac address
Dinget al. [66]		Mac address
G. Li *et al.* [36]	Based on packet headers	NSH
Mehmeri et al. [59]		NSH
Abdelsalam et al. [63]		Segment Routing Header
Cerrato et al. [75]	Based on SDN Switch	Xdpd Switch
Zhang et al. [82]		OpenFlow1.1 Metadata

A. Header-Based SFC Approaches

• The Software-Programmed Networking Operating System (SPNOS) [59] developed by Mehmeri *et al.* is an NFV/SDN SFC solution over packet/optical networks. SPNOS implements Object-Oriented Virtualization [77] for offering service management flexibility to individual tenants. This solution treats services as dedicated virtual network objects that are controlled by an Arbiter. The architecture of SPNOS is composed of a service plane and a network plane. The service plane is implemented based on TOSCA NFV template [78] and is managed by the OpenDaylight controller. The network plane is composed of packet/optical devices.

Traffic steering: this solution is based on a personalized version of the OpenDaylight SDN Controller that implements NSH protocol for SFC configuration and traffic steering.

Advantages: the main advantage of this approach is the focus on optical networks and deployment in multi datacenters.

Limits: the approach relies on NSH and inherits the advantages and limits of header-based traffic steering. NSH requires other tunneling protocols to encapsulate NSH packets, thus encapsulation/decapsulation actions. Moreover, NSH and the support of tunneling protocols add complexity to the approach deployment and can introduce communication delay and overhead.

• Another SFC solution is proposed in [36] by Li *et al.* in the context of mobile edge. This solution aims to provide security SFs for mobile users. A cloud infrastructure implements security SFs and SFC service plane. The architecture is implemented in a cloud and SDN environment, using OpenStack [79] and Pox SDN controller [80], to dynamically instantiate and manage virtual services and program data-plane devices. The authors also present an algorithm for SFC composition based on a fuzzy inference system.

Traffic steering: this solution implements NSH protocol [42] and deploys the SFC architecture [2] (RFC 7665). Thus, the packets are encapsulated with NSH header for service and path identification. A CL that first filters traffic and adds NSH header to packets delimits the SFC domain. The CL is provided with the topology and SFC information by the SDN controller that manages the high-level SFC operation. The forwarders decode packets with NSH header and deliver traffic to the accurate security SFs. In case the SFs are not NSH header aware, a proxy is implemented to translate an NSH packet to an IP packet for the SF and inserts back the header after leaving the SF.

Advantages: the specificity of these approaches is their deployment context; focusing on security service function chaining for mobile computing and moving security SFs to the cloud in order to make the services close to the user side.

Limits: like the previous SFC approach, it relies upon NSH protocols and takes its advantages and limits.

• In the same way, an implementation of VNF chaining through segment routing in a Linux-based NFV Infrastructure is proposed by Abdelsalam *et al.* [63]. It is an Open source SFC solution based on IPv6 Segment Routing header. The architecture of SR-SFC builds on the design of [62]. This

solution is deployed in an NFV/SDN infrastructure. This approach proposes a Linux NFV host that supports multiple VNFs, implementing a Linux kernel module for inner and outer host traffic steering.

Traffic steering: At the ingress edge router, a SRH aware CL classifies traffic flows and inserts SR headers. SRH is composed of a list of IPV6 addresses which is relative to the intermediate VNFs corresponding to the accurate service chain for the flow. Intermediate IPv6 or IPv6/SR routers forward the packets to the proper NFV nodes hosting the VNFs. The Linux based NFV node supports a SR/NF connector, which steers the packets through the VNFs. The connector plays the role of an inner-host SRH forwarder and proxy. As the packets are steered based on the information stored in the SR header, the VNFs are expected to decode the new header. If a VNF is not SRH aware, an SFC proxy is implemented to encode and decode the SR header.

Advantages: as an open source solution implemented in Linux, it encourages its usage and improvement by other researchers and engineers. The inner NFV host traffic steering permits to further take advantage of virtualization to reduce communication delay compared to the regular physical links. The authors accentuate an efficiency in CPU usage. Also, this solution can coexist with an NSH scenario.

Limits: in this work, only the CPU usage is considered, however, other important efficiency metrics are neglected, namely the communication overhead, latency, and delay. Moreover, the encapsulation, decapsulation and reclassification of traffic after visiting an un-aware SR VNF can introduce delay and deployment complexity.

B. Tag-Based SFC Approaches

• The Segmented Proactive Flow Rule Injection (SPYRI) scheme proposed in [70] is an SDN-based solution for SFC. The architecture consists of an SDN controller that handles policy elements, the flow rule injector functionality and the SPFRI engine, and many OpenFlow switches. This solution implements the concept of segmented service function chains, which decomposes the service path of a given chain into segments. These segments are portions of the service path between the switch connected to a given middle box MBox and the switch to which the next MBox is connected. The reason for the segmentation concept is to make it easier to find the next MBox within an SFP.

Traffic steering: in this solution, the controller assigns an SFC ID for different types of flows – all packets are tagged with the corresponding SC-ID. A double tagging system is implemented as an inner tag for the SC-ID and an outer tag that identifies the next SF node to be crossed by the flow. VLAN IDs are used for marking packets.

Advantages: SPFRI uses a tagging scheme based on VLAN and MPLS. The tags do not add much bits to the packets compared to the SFC headers (described in Section III). Moreover, this approach does not require modifications of MBoxes and maintains consistency after traversing MBoxes that might alter packets.

Limits: MBoxes are required to configure a VLAN interface for each SC. This can limit the scalability and flexibility of the approach. Thus, MBoxes might not support an incremental number of service function chains. Moreover, the use of multiple fields (i.e., VLAN, MPLS, QOS, and PCP) can interfere with other networking configurations.

• Also, Simpson proposed in [52] an SDN controller for the context-aware data delivery of dynamic service chaining in the context of Next Generation Service Overlay Networks (NGSON). This approach assumes that SDN and NFV can be used to design and operate SFCs based on the application requirements. The SDN controller uses a dedicated interface to retrieve application requirements. It consists of a customized Floodlight control plane and a service plane. The latter is a request manager that is directly attached to the application entities. A network server connects the service plane with the SDN controller.

Traffic steering: for this solution, SFC traffic steering is performed based on VLAN IDs. Mainly, a unique VLAN ID identifies each MBox instance. Forwarding decisions are made using VLAN ID tags with flow header fields, which are the ingress port, the source IP address, and the destination IP address.

Advantages: above the SDN and NFV technologies, this approach considers the deployment in NGSON context. This enables context-aware and adaptive service functions composition based on network performance metrics.

Limits: this approach does not consider the opaque MBoxes that alter packets. Thus, context information can be lost. Moreover, an incremental number of traffic classes results in increasing the number of flow rules in a switch. Actually, the number of flow rules is conditioned by the TCAM capacity in switches and presents a common challenge among the wide range of SFC approaches and programmable switches in general.

• Conjointly, the SDN-based Source Routing approach proposed by Abujoda *et al.* [71] is a solution designing and operating SFC in datacenter environments. This solution aims to encode the list of SFs of a service chain in a header to get over the need to store the forwarding state in switches. The proposed architecture relies on an SDN controller, and two types of OpenFlow switches: The Source Routing Encapsulation switch (SRE) and the Source Routing switch (SR).

Traffic steering: The controller constructs SFPs by switching output ports, and then encodes the SFP into the packet header's destination MAC address, the VLAN ID, and the MPLS label. SFC metadata is inserted in packets by the SRE switch, and the SR switches forward packets according to the SFC path information.

Advantages: The purpose of this approach is to avoid storing the forwarding state in switches by encoding the SFC path in the packets.

Limits: Encoding the path using multiple tags (i.e., via VLAN, MAC and MPLS) increases the packet size; it can exceed the maximum segment size. This results in fragmentation issues. Moreover, the solution is suitable for datacenters

with a small number of hops in the paths. It is not designed for ISP or Enterprise networks with a high number of hops.

• Furthermore, Qazi *et al.* present SIMPLE [67] for SIMPLE-fying Middlebox Policy Enforcement using SDN. It is another architecture for achieving policy-based and dynamic MBox composition and resource management. The approach is based on extended programmable switches to support service chaining and controller modules. The latter permits to balance the load across MBoxes to ensure their mapping, and to generate forwarding rules.

Traffic steering: As for the forwarding logic, it is based on tags, encoded as VLAN IDs or TOS fields, and inter-switch tunnels, to reduce the number of forwarding entries by using compact forwarding tables.

Advantages: SIMPLE considers balancing the load between MBoxes based on the actual traffic load to prevent overload. This solution considers also reducing the forwarding state by employing tunnels between switches and using compact forwarding tables. Although this approach relies on packet tags that could be lost with MBox modifications, a scheme to track changes is implemented to avoid losing context information.

Limits: a high accuracy matching is needed for tracking packet changes after visiting an MBox. Also, packets are sent to the controller before and after being processed by an MBox; this may introduce additional computational and communication overhead.

• Inspired by SIMPLE [67], another architecture is proposed by Fayazbakhsh *et al.* [68]. This architecture is based on controller modules to configure programmable switches and MBoxes. The main characteristic of this approach is its ability to enhance MBoxes to be part of the traffic steering operations, by enabling them to generate and/or consume tags.

Traffic steering: the forwarding decision is based on packet tags that can be encoded in different packet fields, such as the Differentiated Service field (DS), as part of the TOS IPv4 field, or using a VLAN ID. The proposed prototype embedded tags in TOS. These tags can be either generated by an MBox or by the controller. The latter installs rules in the switches; the forwarding is based on the generated tags.

Advantages: this approach involves MBoxes in the forwarding operation. MBoxes insert tags into the packets to communicate context information that can be used by switches. Involving MBoxes enable the network diagnosis and checking.

Limits: in order to enable MBoxes to tag packets, a modification of MBoxes is required. It is not flexible for a network operator to extend all network MBoxes. Moreover, the modification of packets in each MBox introduces overhead.

• Li *et al.* [64] present another different SFC solution, integrating the technologies of SDN, NFV, and the Path Computational Element (PCE), dubbed PCE-SFC. The architecture of PCE-SFC is based on a PCE architecture [81] and is composed of a control plane and a forwarding plane. The controller, which is the main component of the control plane, consists of two modules: a module for service function registration and inquiry, and a path computation module. The forwarding plane consists of gateway routers that function as a CL and thus encapsulate the SFC path information to the packet's headers. The controller provides the CL with SFC paths and mapping information about the topology. Another two types of routers are deployed: the first service-providing router which the service functions are connected to, and the data-delivery router that is responsible for the traffic steering.

Traffic steering: the CL filters and encapsulates the packet headers with SFC path information and the data-delivery routers that decode the header, and then forwards the packets to the next destination in the chain. This approach introduces an IPv4 option to convey the SFC header, composed of a list of services and path identifiers.

Advantages: this approach is based on a PCE architecture; it considers network parameters for path computation. An algorithm for service chains composition is proposed based on link delay and packet loss. Considering such parameters in SC composition improves the network performance.

Limits: the SFC path is encoded in the packet header (new IPv4 option) as a set of identifiers. Yet, the size of the header may increase as per the number of traversed nodes and SFs. Thus, longer chains and higher numbers of hops can introduce overhead.

• Ding *et al.* [66] proposed another SDN-based solution that facilitates the SFC design and operation. In this architecture, to customize SFC features, the management plane benefits from the abstraction of the infrastructure provided by the two other planes, namely the control plane and the data plane. To enable the automatic configuration of the data plane devices, the control plane contains different controllers, namely, the policy controller, the SDN controller, and the NFV controller. The OpenSCaaS orchestrator, located in the management plane, defines the policy context, and the controller translates it into a set of rules that are sent to the data plane elements. Finally, the data plane consists of the networking devices and Software Defined - MBox (SD-MBox) that includes the SFC CL and the involved switches.

Traffic steering: OpenFlow switches forward traffic based on the source MAC address. The MAC address is used as an SFC identifier (SC-ID) processed by switches to make forwarding decisions.

Advantages: the approach reduces the forwarding state, using an SC identifier (i.e., source Mac address); this permits to aggregate micro flows of an SC into one macro flow. Also, the authors consider the reactive rule provisioning for CLs.

Limits: the problem of packet modifications by MBoxes that result in losing the context information applies to this approach. The MAC address can be altered by an MBox or by a router. Thus, it is not clear how the context information can follow the journey of packets after traversing routers.

• The work in [65] also introduces an SDN-based approach for SFC. The proposed solution implies a three-layer architecture: the application layer, the control layer, and the infrastructure layer. The application layer entails an SFC controller module that is responsible for carrying out business application requests. The control layer processes the network related information and contains the SFC Router module. Finally, the infrastructure layer is where the devices reside.

Traffic steering: Traffic steering for SFC is inspired by the work in [82], as it relies on the MAC address and ports of

edge routers, while IP addresses are used to identify users and serve for traffic classification.

Advantages: the approach relies on L2-L3 for traffic identification, which avoids an increase in packet size. Also, network isolation is used to identify SCs even when MBoxes modify packets headers.

Limits: the creation of dedicated SF instances for each SC can raise resources and management challenges when the network scales out.

• Finally, an SDN-based SFC mechanism is proposed by Trajkovska *et al.* [72]. It is a NFVI-cloud eco-system over an SDN solution for SFC, based on the following open source implementations: T-nova [83], Opendaylight [84], OpenStack [85], a customized version of Netfloc [86], and other APIs. The SFC operation is initialized by the orchestrator, creating VNFs and triggering the WAN infrastructure connection manager API (WICM) with clients' requests for an ID space; an ID space using VLAN IDs for clients' isolated traffic management. WICM tags users' traffic with VLAN IDs. WICM API is also responsible for traffic direction to the accurate NFV-PoP (SFC Pop). Mainly, Netfloc is responsible for traffic steering and vTC [87] for traffic classification.

Traffic Steering: Traffic steering relies on L2-based OpenFlow rules, processed by SDN switches that are configured by OpenDayLight SDN controller. Netfloc API achieves traffic steering through the VNFs based on rewriting MAC addresses. The SFC ID and the VNF ID are encoded in the destination MAC address. The VNF ID is decreased after each VNF visit. The original MAC address is saved in Netfloc and rewritten back in the final hop iteration before reaching the packet destination.

Advantages: the main advantage of the described solution is the use of open source tools and the development of open source components that can be upgraded and improved by a community. Also, this solution considers data-path redundancies.

Limits: the described solution relies on VLAN IDs to identify users traffic, and MAC addresses for identifying VNFs. Scalability issues arise when using VLAN IDs; first, it can limit the number of users, and second, it requires configuring virtual interfaces for these VLANs.

C. SDN Switch-Based SFC Approaches

• A different user-specific approach for service chaining is presented in [75] in the form of an SDN enabled node that permits the connected users to create network service graphs. The proposed architecture is based on a global orchestrator, a node orchestrator to instantiate network functions, virtual switches, and a controller per each user. The global orchestrator leverages the high-level information, such as user profiles, service graphs and the description of network functions forwarding graphs.

Traffic steering: the forwarding is based on the Extensible Data Path daemon (xDPd). The specific Logical Switch Instance is created for each user to ensure network function isolation and the required forwarding rules are provided by the per-user controller.

Advantages: SDN-enabled network node is a different approach that is user specific. Users are connected to dedicated virtual switches that permit service chaining in an isolated network. Isolation permits traffic steering while omitting the use of headers or tags, and it is not affected by MBoxes that alter packets.

Limits: scalability and management issues can rise with an incremental number of users, as the number of physical ports, through which a user is connected to the node, is limited.

• Traffic steering for software inline services and forwarding (StEERING) [82] is one of the first solutions for dynamic and programmable SFC. The traffic steering system stands on the SDN controller and OpenFlow switches. The controller contains two modules to install flow entries into the switches and handles the service placement. As for the switches, they are responsible for traffic classification and steering.

Traffic steering: the solution is based on Layer 2 and Layer 4 contents of packets and the direction of traffic is known from the ingress port. The SFC metadata is encoded as an OpenFlow 1.1 metadata field.

Advantages: the proposed approach allows for fine granular subscriber and application policies. Using multiple forwarding tables reduces the number of flow rules. Moreover, a service selection algorithm is proposed to optimize network performance.

Limits: the approach does not handle the case where MBoxes modify packets, which can lead to misleading policies.

D. Other Approaches

In this section, we present other SDN-based SFC approaches that could not be classified, mainly because of the lack of information about the traffic steering operation used.

• Csoma *et al.* proposed in [88] the Extensible Service Chain Prototyping Environment (ESCAPE). This is another approach based on tools such as Mininet, Click, NETCONF, and Pox, as well as an orchestrator. NETCONF is responsible for mapping Click VNFs, while the Pox controller handles traffic steering. The architecture is decomposed into three layers: the service layer, the orchestration layer, and the infrastructure layer. The service layer handles service requests, while the orchestration layer maps VNFs to physical resources. The virtual and physical resources represent the infrastructure layer.

Traffic steering: As for the traffic steering logic, a dedicated module hosted in the Pox controller handles ESCAPE. However, there are no specific information about the traffic steering method implemented in ESCAPE.

Advantages: ESCAPE provides a ready environment for researchers to develop and test their proposals. It provides the basic SDN, NFV and emulation tools and components.

Limits: the update rate is slow and the documentation needs to provide further details.

• A deployment of SFC in the industrial network is presented in [35] by Fysarakis *et al.* This solution focuses on a Wind park security use case, aiming to deploy security services in a flexible and programmable way. Mainly. SDN/NFV

technologies coupled with SFC enable the flexibility of service deployment, especially for pricing and sophisticated security SFs.

Traffic steering: for SFC management and traffic steering, the proposed architecture deploys a CL to define the legitimacy of external traffic flows and adds packet headers for achieving traffic steering. An SDN controller provisions the CL with SFC information to encode packet headers and SDN forwarding elements with flow rules to forward packets. This solution uses the OpenDaylight controller to manage Service functions based on the SFC-ODL project [89]. However, no thorough information on the traffic steering operations is mentioned.

Advantages: this approach is ready to deploy for industrial networking infrastructures. It also considers the security of SFC mechanisms which is important in a production environment. Moreover, it supports fine granularity policies, for per application or tenant profiles.

Limits: The approach does not provide details on the SFC mechanisms used; it is not possible to know the limits of the proposal. Moreover, no performance evaluation is carried out.

E. Summary and Insights

The smooth operation of SFC depends on the underlying traffic steering method and on the properties of the context for which that SFC is deployed. Specifically, the discussed SDN-based SFC approaches have nearly the same architecture, principally inspired from the SDN architecture. The main difference is in the deployment context and the algorithms and the high-level applications for SFC. The properties of the SFC deployment context can define the appropriate traffic steering method. The applications and algorithms accordingly achieve different goals (e.g., shortest path calculation, optimal path based on context information, and chains composition).

The various traffic steering methods used in SDN-based SFC approaches reflect the infrastructure contexts and goals. We believe that the header-based SFC solutions are suitable for deployment in wide geographic infrastructures, across datacenters, mobile and optical networks, while tag-based SFC schemes can be deployed in datacenters. It is worth noting that routers usually modify L2/L3 headers and thus the SFC forwarding information can be lost across data-centers. The SFC based on programmable switches is agnostic to the deployment context (i.e., the infrastructure's data plane) since the traffic is classified each time it crosses an SFF.

The design or choice of the accurate traffic steering method also depends on the operator's vision, requirements, and priorities. Some approaches accept that the header-based traffic steering introduces support complexity within SFs and SFFs at the cost of reliability and scalability, while the tag-based traffic steering can achieve a ready to deploy and less complicated SFC at the cost of manipulation of tags by opaque SFs or other devices. Other operators can choose to not mangle packet headers, rely on programmable switches, and accept complicated flow matching or repeated classification.

V. Qualitative Evaluation of SDN-Based SFC Approaches

While several topics under the scope of SFC are actively studied (e.g., service decomposition, service modeling, and traffic steering), the efficiency of the traffic steering methods used in each SFC approach is fundamental and influences its overall performance. For this purpose, we choose to discuss the different traffic steering methods and investigate the efficiency of the aforementioned SDN-based SFC approaches. In this section, we provide a qualitative efficiency evaluation of the traffic steering method and discuss its impact on a comprehensive list of SDN-based SFC solutions. The evaluation relies on the taxonomy, described in Section III, to determine how the different traffic steering methods can impact the scalability, the flexibility, the overhead, and the deployment cost of an SFC solution. Moreover, we present a benchmarking of the validated techniques and environments of the studied SFC approaches.

A. Efficiency Evaluation Criteria and Metrics

In this paper, efficiency refers to the metrics that are necessary for assessing the performance of an SFC architecture. To do so, we set four efficiency criteria: scalability, flexibility, expected deployment cost, and overhead.

1) Deployment Cost: To define whether the implementation of an approach is costly, we investigate the following information:

- NFV deployment support.
- The use of resource optimization techniques.
- The non-generation of additional configurations.
- Ability to reuse the available SFs.

2) Flexibility: Flexible SFC solutions should also take into account several metrics:

- The deployment of an SDN controller should ensure network visibility, programmability, and thus, flexible service chain composition.
- The NFV orchestrator should enable the dynamic instantiation of VNF or virtual service functions. The VNF management should improve flexibility.
- Support of fine-grained policies, e.g., per user and application granularity.
- Avoidance of static MBox modification or configuration.
- Support of both proactive and reactive flow rule provisioning.
- Support of multiple SDN protocols for communication between the control plane and data plane, e.g., OpenFlow [43], Netconf [44], and LISP [90].

3) Scalability: For the deployment in large-scale networks, we check that the studied SFC solutions consider:

- Reducing the number of forwarding rules.
- Reducing flow rule entries in the switches.
- Encoding the SFC context information or metadata in specific tags, reducing the need for storing complex forwarding information in the memory of switches.
- Using compact SFC Metadata, thus a maximum of information can be stored in minimum header or packet field.

- Expansion of the number of service chains, regarding the length of the service chain field; in other words, the maximum number of chains that can be encoded in the field.

4) Overhead: The Overhead generated by certain solutions must be taken into consideration. We propose some metrics to determine if an approach results in an acceptable overhead or not:

- The order of control communication between the controller and the switches.
- The packet fields reconstruction overhead.
- Network and SFC encapsulation overhead.

B. Deployment Cost

The first considered criterion is the expected cost of deploying a new technology. Thus, reducing the OPEX and CAPEX must be taken into consideration. This analysis does not consider the cost induced by the introduction of a new controller, the software licensing modules, and the integration-relevant cost.

SDN enables programmable and controllable configuration of networking devices, which reduces the time to market and the cost of configuring networking devices statically. Furthermore, one of the objectives of SDN and NFV is the separation of the control and data forwarding planes in networking devices and service functions, thus enabling the use of software over generic hardware, which reduces investment in hardware. Likewise, the claim for NFV is that it would help to reduce the investment cost of acquiring new hardware. Indeed, resource optimization plays a significant role in reducing the deployment cost and the investment cost, which avoids over-provisioning.

According to the deployment cost, Ding *et al.* [66], Martini *et al.* [69], and Csoma *et al.* [88] considered the VNFs deployment. The deployment of Software Defined Mboxes (SD-Mboxes) and virtual Mboxes reduces the investment cost as much as the deployment of StEERING [82] and the proposition of Ding *et al.* [66] do. Also, the configuration cost must be also taken into consideration. However, SPFRI [70] and the position paper [65] require an interface configuration per MBox, which is costly and error-prone. Despite the solutions described in [65] and [70], the other solutions permit the reusability of SFs.

One of the objectives of service chaining is to re-use SF instances, which enables the instance to be part of multiple service chains. However, the approach described in [88] dedicates SF instances per service chain and considers an SF instance per each user [75]. This could enhance flexibility at the cost of creating additional expenses.

Consequently, the deployment cost of the studied solutions is optimal according to the evaluation metrics. However, the header-based solutions that require SFs support or proxies may induce additional investment, especially in large-scale networks. Table IV provides a quick reference to the estimated deployment cost of the studied SDN-based SFC solutions.

Recommendation: the deployment of an SFC solution based on SDN and NFV reduces the deployment cost. Thus, implementing algorithms for optimizing resources and employing virtual SFs or VNFs is highly recommended [91]–[94]. Moreover, the runtime cost of SFs has to be studied as a dynamic scaling based on the actual experienced load which can save resources. Also, while designing the routing logic for traffic steering, the load on SFs should be considered to avoid allocating new resources for SFs while other SFs can be used by rescheduling paths based on the runtime load of SFs. This improves the quality of service and avoids scaling out and allocating further resources while using the existing resources.

C. Flexibility

The flexible management of SFC within an SDN architecture is necessary, especially for large-scale networks where tedious tasks of configuration take place. The control plane in SDN enables a global view of the infrastructure, a control of devices as well as their programmability so that SFC can be created on-demand. Mainly, the SDN controller promotes SFC flexibility by administrating traffic steering mechanisms and programming the switch flow tables, usually used as SFFs in the context of SFC [72]. The studied solutions inherit a certain flexibility level from being deployed in an SDN environment and administering the traffic steering based on a controller. Effectively, SDN technologies enable the topology independence by separating the software intelligence from the hardware (e.g., traditional routers). Thus, they facilitate a flexible deployment of solutions on generic hardware (e.g., running virtual switches), which decreases the cost of resource provisioning and avoids the configuration complexities.

Furthermore, the deployment of SFC in an NFV environment enhances the flexibility as well. Approaches such as the ones proposed by Martini *et al.* [69] and by Trajkovska *et al.* [72] deploy an orchestration layer for the dynamic provisioning and management of virtualized SFs and physical resources. Other schemes propose their extensible network orchestrator such as Csoma *et al.* [88].

The support of SFC at the user or application granularity enhances customized policies and flexibility at the cost of raising scalability issues [69]. Some of the studied solutions do not require any MBox modification and can coexist with non-programmable or SDN-enabled switches. However, other solutions need MBox support such as the scheme proposed in [65] that requires minimal configuration on MBoxes, which may incur negligible overhead. Nevertheless, it creates additional tedious tasks of configurations, and adding or removing MBoxes can lead to an inconsistency within the infrastructure and SFC operations. Fayazbakhsh *et al.* [68] assume modifying the source code of MBoxes to make them part of the SDN operation and to communicate with the controller, at the cost of having to adjust the source code of each new service function.

Similarly, the packet header-based techniques require modifications to the switches and the controller to support the headers. Moreover, SFs need to support the headers or to be attached to proxies that ensure header insertion and removal from the SF. Consequently, the required support and proxies

TABLE IV
TAXONOMY OF DEPLOYMENT COST ANALYSIS OF THE STUDIED SDN-BASED SFC SOLUTIONS

Steering technique	SDN based SFC solutions	The none modification of MBox source code	The non-generation of additional configurations	NFV deployment support	Ability to reuse the available SFs
Packet tags	Fayazbakhsh et al. [68]	×	×	×	✓
	Trajkovska *et al.* [72]	✓	✓	✓	✓
	Ding et al. [66]	✓	✓	✓	✓
	Blendin et al. [65]	✓	×	×	×
	Martini et al. [69]	✓	✓	✓	✓
	Qazi et al. [67]	✓	✓	×	✓
	Abujoda et al. [71]	✓	✓	×	✓
	Pawar et al. [70]	✓	×	×	✓
Packet headers	Mehmeri et al. [59]	×	N/A	✓	✓
	Abdelsalam et al. [63]	×	✓	✓	✓
	G. Li *et al*. [36]	×	✓	✓	✓
	T. Li et al. [64]	✓	✓	✓	✓
Programmable switch	Zhang et al. [82]	✓	✓	×	✓
	Cerrato et al. [75]	✓	✓	✓	×
Others	Fysarakis et al. [35]	N/A	N/A	×	✓
	Csoma et al. [88]	✓	✓	✓	✓

increase the deployment complexity of a given solution. Furthermore, SPFRI [70] does not consider reactive flow rule provisioning unlike in the work of Ding *et al.* [66] and Qazi *et al.* [67]. Hence, proactive rule provisioning reduces the flexibility to define real-time policies.

All the studied approaches use the OpenFlow protocol for exchanging information with data plane elements. However, no support for other protocols, such as NETCONF [44], has been taken into consideration. From the flexibility perspectives, an efficient SFC approach should not be limited to OpenFlow environments. Table V provides a reference to the flexibility of the studied SDN-based SFC solutions.

D. Scalability

SFC is a solution designed to address deployments in large-scale networks, coping with hundreds of SCs, dynamic SF instances selection and placement, and thousands of classification rules. There is no benefit in enabling SFC techniques in networks with few SFs or networks with constraints on the dynamic instantiation of services. It is advantageous to use compact metadata to encode SFC context information for reducing processing and memory requirements in switches. Indeed, storing context information on the limited memory of switches is challenging. The challenge is to store the maximum service chains in the minimum storage possible. Solutions, such as the ones proposed by Simpson [52], Ding *et al.* [66], Qazi *et al.* [67], and Pawar and Kataoka [70], used packet tags such as VLAN IDs for encoding SC information, which is a more favorable and compact field of 8 bits. However, the packet header-based traffic steering that encodes multiple fields such as NSH increases the header size and thus the overall packet size, which may exceed the Maximum Transfer Unit (MTU).

Network encapsulation protocols such as VxLAN insert additional headers in the packets, resulting in additional packet size. Additionally, SFC encapsulation for tunneling between the SFC elements increases packet size as well, especially the packet header based solutions that use new packet headers for SFC encapsulation. As a result, with the increase of traffic size, the SFC encapsulation and network encapsulation limits the scalability of such approaches in large-scale networks.

Henceforth, another metric should be considered which consists of the ability to expand the number of service chains with the type of field or header used. In the case where the VLAN ID is the SC identifier, the metadata field is compact. However, there is a limitation when increasing the number of service chains as per large-scale networks. As mentioned in [95], a recommendation of SC field of 32 bits fulfills the need to deploy service chains in very large-scale networks. However, even though the packet header based methods permit an expansion space to encode a high number of SCs, the scalability issue arises due to the increased packet size as stated earlier.

To address the scalability issue, some optimization techniques have been investigated to reduce the number of rules to be installed in switches as proposed in StEERING [82]. StEERING is based on multi-table SDN switches. Similarly, Qazi *et al.* introduce in [67] compact flow tables and use switch-tunnel tables to reduce the number of flow entries, for the mapping between intermediate switches that are not directly connected to MBoxes. Martini *et al.* also implemented optimization techniques in [69] to reduce the number of flow rules, using a class-based forwarding technique. Another possibility to reduce memory constraints is to obviate storing the SFC forwarding state in switches. Most of the studied approaches encode SFC information in the packet fields or headers and thus significantly reduce the size of flow tables.

Other Optimization techniques are introduced in [68] by Fayazbakhsh *et al.* to reduce the number of necessary header bits, temporal reuse, special reuse and coarser tags. However, when switches have to deal with thousands of VLANs, scalability issues arise. Table VI compares among the studied SDN-based SFC solutions in terms of scalability.

Recommendations: For scalability, we encourage encoding the mandatory information such as the SC identifier and avoid encoding information that could be derived from SFFs. Additionally, the cost relevant to the memory and processing power when encoding SFC information in packets should be considered. In other words, encoding optional information

TABLE V
TAXONOMY OF THE FLEXIBILITY ANALYSIS OF THE STUDIED SDN-BASED SFC SOLUTIONS

Steering technique	SDN based SFC solutions	Deployment of a SDN controller	Deployment of a NFV Orchestrator	The none modification of Middlebox source code	Avoiding per instance modification or configuration
Packet tags	Fayazbakhsh et al. [68]	✓	×	×	×
	Trajkovska *et al.* [72]	✓	✓	✓	✓
	Ding et al. [66]	✓	✓	✓	✓
	Blendin et al. [65]	✓	×	×	×
	Martini et al. [69]	✓	✓	✓	✓
	Qazi et al. [67]	✓	×	×	✓
	Abujoda et al. [71]	✓	✓	×	✓
	Pawar et al. [70]	✓	✓	×	×
Packet headers	Mehmeri et al. [59]	✓	✓	✓	✓
	Abdelsalam et al. [63]	✓	✓	×	✓
	G. Li *et al*. [36]	✓	×	×	✓
	T. Li et al. [64]	✓	×	✓	✓
Programmable switch	Zhang et al. [82]	✓	×	×	✓
	Cerrato et al. [75]	✓	✓	✓	✓
Others	Fysarakis et al. [35]	✓	×	N/A	N/A
	Csoma et al. [88]	✓	✓	✓	✓

TABLE VI
TAXONOMY OF THE SCALABILITY ANALYSIS OF THE STUDIED SDN-BASED SFC SOLUTIONS

Steering technique	SDN-based SFC solutions	Compact metadata	Optimization to reduce forwarding state	Service chain number expansion
Packet tags	Fayazbakhsh et al. [68]	✓	✓	×
	Trajkovska *et al.* [72]	✓	N/A	N/A
	Ding et al. [66]	×	✓	×
	Blendin et al. [65]	×	×	✓
	Martini et al. [69]	✓	✓	×
	Qazi et al. [67]	✓	✓	✓
	Abujoda et al. [71]	×	✓	×
	Pawar et al. [70]	✓	N/A	×
Packet headers	Mehmeri et al. [59]	×	×	×
	Abdelsalam et al. [63]	×	×	✓
	G. Li *et al*. [36]	×	×	×
	T. Li et al. [64]	×	✓	×
Programmable switch	Zhang et al. [82]	✓	✓	✓
	Cerrato et al. [75]	N/A	✓	×
Others	Fysarakis et al. [35]	N/A	N/A	N/A
	Csoma et al. [88]	N/A	N/A	N/A

in packets increases the packet size and limits the scalability. Similarly, calculating the paths by the switches or storing the SFC information in the switches can lead to scalability limitations due to the incremental processing and memory costs in the controller and devices. As recommended in [95], an SC field of 32 bits fulfills the need of deploying service chains in very large-scale deployments.

It is true that NFV enables the dynamic allocation of resources for SFs, yet the dynamic adjustment of resources for SFs, based on runtime requirements to scale in and out, improves the performance of the network and permits to save resources. By increasing resources based on the actual system requirements (e.g., SFs load, user requirements, and differentiated policies), the system can enhance the scalability and elasticity of the network.

E. Overhead

In SDN architectures, the amount of signaling traffic to be exchanged between the data plane and the controller can raise performance and scalability issues, especially, when the control communication overhead is incurred by the exchange between controllers and switches.

Encoding SFC metadata in packet headers or tags reduces communication overhead between switches and the controllers by using tag-based rules. In other words, SFFs, usually SDN enabled-switches, do not have to ask the controllers on how to forward traffic flows. Indeed, the matching is based on the data carried in packet headers or tags. In case of SFFs, this reclassifies traffic to forward packets. This method decreases the communication overhead but increases the processing overhead in SFFs as well as the number of flow rules in the forwarding tables, resulting in memory saturation. Most of the aforementioned solutions are based on encoding some SFC information in packets to enable traffic identification and forwarding. Additionally, a processing overhead may occur in the controllers and SFFs due to header's reconstruction at every MBox and SFF. This operation generates a considerable overhead as well. Moreover, a processing overhead and additional delay may occur in case of traffic steering methods that require packet reconstruction at SFFs and SFs. In spite of the approaches that require packet rewriting, the proposition of

Abujoda *et al.* [71] requires switches to decrement a counter and forward packets to the next SF in the path. This information is already retrieved by the controller and is written in the header. Thus, the processing overhead is reduced, and no header re-writing is required.

Most of the studied traffic steering methods, especially the header-based methods, require network encapsulation operations. Consequently, this leads to an additional processing overhead to encapsulate packets as well as increased packet sizes that may result in considerable delays, especially in large-scale networks. A proposition to reduce the overhead by reducing the communication control overhead using two types of switches is proposed by Blendin *et al.* [65]. A core switch communicates with a controller and initiates the data paths while other switches steer traffic based on the information in the packet headers.

Recommendations: The traffic steering techniques based on headers are more likely to generate overhead than other traffic steering techniques. We recommend the use of existing tags to encode the essential SFC information. The SC identifier and the next SF are enough for an SFF to steer the traffic with a reduced number of forwarding states. Furthermore, reducing the encapsulation overhead, controlling the communication overhead, optimizing the matching processing time and reducing the forwarding state may significantly reduce the overall traffic steering overhead. A tradeoff has to be thoroughly studied to determine the impact of packet size, encapsulation, processing time and resources, memory, and control communication overhead. Table VII compares among the studied SDN-based SFC solutions in terms of some overhead criteria.

F. Validation Techniques Used in SFC Approaches

An important part of studying SFC approaches is to perform a benchmarking of SFC deployment environments to assess the architectural differences and network types as they impact the overall SFC process including traffic steering. In this section, we present a benchmarking of the validation methods used in the studied SDN-based SFC approaches in different environments. Table VIII summarizes the validation methods used by various SFC approaches. It shows that almost all studied prototypes are implemented in small testbeds and small-scale topologies. This is due to the heterogeneous deployments, incompatible architectures, and prototype-specific limitations [72]. In the current study, we use a variety of metrics to ensure the validation of each approach based on its objectives and effectiveness, although common and specific metrics are hard to find for the different assessed approaches.

Hence, in addition to the deployment heterogeneity and the lack of common metrics, SFC approaches are not compared against each other nor tested in large-scale networks (except for two proposals, namely [67] and [68]). The comparison is not feasible because of the difficulty to find specific information in the same environment (see Table VIII). Usually, approaches are validated looking at a baseline system to show the added value of deploying the approach. For example, Martini *et al.* [69] compared a traffic that passes through all SFs (the baseline system in this case) with classes of a traffic that passes through the required SFs.

VI. Discussion and Research Challenges

In this section, we discuss the evaluation conducted in the previous section and outline the ongoing challenges related to traffic steering and SFC.

A. Discussion

Some of the significant advantages of the SDN-based SFC approaches consist in reducing cost and overhead. Thus, coupling SDN with virtualization techniques significantly reduces the investment cost in hardware solutions. Avoiding the re-construction of SFC headers significantly reduces the switches' processing overhead and delay. However, the deployment in large-scale networks requires tackling some of the scalability and flexibility issues. Admittedly, the studied solutions support a management layer for SFs and SFCs at different levels of granularity (e.g., per subscriber or application). This comes at the cost of increasing the number of rules installed in switches or handled by the controllers. The latter impacts the capacity of internal memory, especially with the expensive TCAMs used in switches today. Also, the size of different headers for holding SFC metadata influences the overall deployment of SFC in a large-scale network. Furthermore, the compact metadata reduces the packet size and traffic delay at the cost of limiting the expansion of the maximum number of SFCs. Indeed, a header or field of size β permits a maximum of $(2^{\beta} - i)$ SFCs, where i denotes the number of non-functional values [95]. Thus, when the size of metadata decreases, the threshold of expansion decreases as well as the traffic delay and packet size.

Furthermore, real-time modifications of SFs or SFCs, such as removing an SF, can lead to an inconsistency within the infrastructure. It is worth mentioning that not all the studied approaches support reactive rule provisioning. They promote flexibility despite being error-prone and leading to inconsistency. An investigation regarding communication overhead between the controllers and the switches is an open research issue. Similarly, the internal processing of the controllers and the switches to calculate the destination of packets should be studied.

B. Research Challenges

By reviewing the traffic steering techniques used in different SFC implementations, we outline some research challenges of particular relevance to traffic steering and service chaining. Specifically, this section highlights the challenges related to the quality of service, scalability, management, and security.

1) Quality of Service: One of the critical research areas in service chaining is the quality of service deployment and delivery. The next generation networks require hyper connectivity and the support of high QoS to cope with the ever-growing demands for bandwidth from end-users and industrial critical requirements. In relation with SFC, several research problems directly affect the QoS of the overall SFC services

TABLE VII
TAXONOMY OF THE OVERHEAD ANALYSIS OF THE STUDIED SDN-BASED SFC SOLUTIONS

Steering technique	SDN-based SFC solutions	Minimize control communication overhead	Minimize packet fields re-construction overhead	Encapsulation overhead: network encapsulation and/or additional packet fields
Packet tags	Fayazbakhsh et al. [68]	✓	✓	×
	Trajkovska *et al.* [72]	✓	✓	×
	Ding et al. [66]	×	✓	×
	Blendin et al. [65]	×	✓	×
	Martini et al. [69]	×	×	×
	Fysarakis et al. [35]	N/A	N/A	N/A
	Qazi et al. [67]	✓	×	×
	Abujoda et al. [71]	×	×	×
	Pawar et al. [70]	×	×	×
Packet headers	Mehmeri et al. [59]	✓	×	×
	Abdelsalam et al. [63]	✓	×	×
	G. Li *et al*. [36]	✓	N/A	×
	T. Li et al. [64]	×	✓	×
Programmable switch	Zhang et al. [82]	✓	✓	✓
	Cerrato et al. [75]	×	✓	✓
Others	Fysarakis et al. [35]	N/A	N/A	N/A
	Csoma et al. [88]	N/A	N/A	×

including, SF placement [96], [97], resource allocation, context aware path calculation and adaptation, encapsulation, and optimization of networking resources such as processing and memory.

The optimal service function placement profoundly affects the delivery time of complex services [98]–[101]. The constraints of placement must be customizable according to subscribers' preferences, infrastructure properties, and service delivery time, while considering the generated cost [102]. A technology such as Multi-access Edge Computing (MEC) [96], [97] can be deployed for SFC benefit to enhance the QoS with lower cost. MEC consists of bringing services to network edges to offer low latency and high bandwidth. Using service placement algorithms and MEC technology for SFC may be beneficial for the SFC infrastructure and may satisfy subscribers' needs.

While the service placement problem is critical for the QoS, the resource allocation is not less important. Indeed, these two problems impact each other. Flexible service placement requires a dynamic resource allocation strategy. One of the VNF challenges is the resources sharing in dynamic provisioning, which permits for a VNF to dynamically acquire the required resources. However, the VNFs compete for the global resources. Thus, VNFs can have fewer resources than required, which raises the VNF failure probability [103]. Moreover, the failure in an SF can cause a whole chain failure.

The mapping of service chains to physical resources and the composition of service chain combinations remain a challenge. Several criteria make the composition problem demanding, such as composing service chains based on service constraints, actual networking state, and subscribers' preferences. Also, with the growing demand on networking services, increasing load, and diversity of services and subscribers, the calculated paths for a service chain need to be adapted to the actual network state. Yet, once the paths are calculated for a given SFC, it is an error-prone task to update the SFC path in real time.

Encapsulation schemes are very often used for SDN-based SFC approaches where some encapsulation protocols are not supported by SFs. In this case, some SFC approaches require traffic to go through proxies before and after each SF, which can introduce considerable complexity, overhead and latency. Moreover, due to multiple SFC encapsulations, the packet size increases. This can have considerable effect on the end-to-end delivery time and overall overhead.

Latency and communication overhead, can be also introduced by longer chains (i.e., with more than 4-6 SFs). Additionally, the amount of traffic exchanged between the controllers and forwarding devices should be considered to reduce communication overhead and processing requirements. We noticed that some SFC approaches rely on the controllers to store the forwarding information in order to reduce the forwarding state, at the cost of increasing communication overhead between the switches and the controllers. Also, the consecutive modification of SFC packets headers and tags can increase processing requirements and latency.

In order to support a fine granular SFC (e.g., per application, user, device, or a mix of complex criteria), the forwarding state gets multiplied, according to the granular traffic classification. The growing forwarding state degrades the performance of the switches; a packet's processing time is mainly dominated by flow table lookup [107]. A common challenge in the SDN community is the improvement and optimization of device capacities. The programmable switches, mainly hardware switches, face TCAM memory limits, which may lead to degradation of QoS (e.g., increased latency) or failure. The software switches do not struggle with memory constraints, as the resources depend on the hardware capacities on which the switches are installed. Yet, the processing capacities are better than the virtualized switches [108]. A tradeoff between memory and processing capacities is challenging to retrieve.

2) Scalability: After reviewing the SFC approaches, we found that the primary challenge of SFC is the scalability. Even though it is widely discussed in the literature, there is still a lack of consideration of scalability when designing SFC

TABLE VIII
EVALUATION SETUP FOR STUDIED SDN-BASED SFC APPROACHES

Approach	Evaluation	Tools	Scenario	Metrics	Comparison
Fayazbakhsh et al. [68]	-Testbed emulation	POX SDN controller, Mininet, MBoxes, KVM	1) Small testbed to access the internet 2) Large pop-level ISP topology: Topology from rocketfuel [113]	MBox overhead, Controller scalability to handle MBox queries, Number of nodes and switches, end-to-end overhead	- Baseline comparison -Differentiating topologies
Ding et al. [66]	-Testbed	Iperf client & server, OpenvSwitch, Clickos	Small typical Gi-LAN testbed	Requests speed, requests size	Packets generated by corresponding received packets
Blendin et al. [65]	-Testbed Emulation (proof of concept)	Foodlight SDN controller, Mininet, NEC PF5240 OpenvSwitch	N/A	N/A	N/A
Martini et al. [69]	Testbed	Service Oriented Controller (Not specified), OVS switch, Mboxes	Small Enterprise network	Path Setup Time (Request-Response), Round Trip Time by packets, number of Flow entries installed in switches	Class-Based With Baseline
Qazi et al. [67]	- Emulation in Emulab -Emulation in Mininet -Trace-driven simulation	Pox SDN controller, OVSswitch, Click modules as Mboxes, Iperf	Operator Network topologies from RocketFuel [113]: Internet 2, Geant topology	Time to install flow rules, total communication overhead, maximum load on MBoxes	Differentiating topologies
Abujoda et al. [71]	-Testbed in Emulab -calculation: flow table as input	Pox, click modular router, Netfpga based packet generator, Oprofile	1) Fat-tree topology 2) Input data for calculation: A forwarding table of 380k entries.	Flow arrival rate, Per flow setup time, communication and control overhead, Computational requirements	Customized switches compared with rule-based forwarding switches.
Pawar et al. [70]	-Proposition: not implemented	N/A	N/A	N/A	N/A
Zhang et al. [82]	-Testbed: proof of concept -Emulation	Nox SDN controller, customized SDN Switch witch ezchipnp4 network processor, Iptables, Apache server, AAA server	1) small testbed: 2)emulation based on unsampled packets collected from European Municipal Network	Number of rules, number of subscribers , CPU utilization, number of SFs, latency	Implemented switch with Pswitch
Cerrato et al. [75]	Proposition	Xdpd SDN switch, NFV orchestrator, Network functions, DPDK framework, Docker container	N/A	N/A	N/A
Trajkovska *et al.* [72]	Testbed	Pica8 SDN switch, Opendaylight SDN Controller, OpenStack, Netfloc	Small testbed: operator network	Throughput, -packet loss, latency, jitter	No comparison
T. Li et al. [64]	Testbed	OpenDayLight SDN controller, OVS switch, VirtualBox, Apache Server	Small testbed: broadband network [114]	Number of SFs, end-to-end path delay, response time of the controller	Comparison with recommendations in [95]
G. Li *et al.* [36]	-Testbed -Mathematical calculation	OpenStack, Pox SDN controller, OVS	Small Testbed: Mobile-edge network	Processing delay, CPU usage and execution time of SFs	Comparing the proposed algorithm with Saw algorithm mathematically.
Fysarakis et al. [35]	Testbed: no evaluation	OpenDaylight SDN Controller, SCADA IDS, Pfsense, EWIS honeypot, Xnmap..etc.	Small testbed: WindPark network	N/A	N/A
Mehmeri et al. [59]	Testbed: Emulation	Mininet, OVS switch, Line-OE for emulating optical nodes, OpenDayLight with NSH support, Tosca template .etc	Small testbed: Packet/optical network	SFC setup time, total setup time	No comparison
Abdelsalam et al. [63]	Testbed	VirtualBox, Linux Netfilter, Srext, Vargant..etc	Small testbed	Number of flows, number of VNFs, generated packet rate, CPU utilization, packet loss ration	Concept comparison with NSH [42]

solutions. Especially, encoding the SFC information in small fields may work for small scenarios, but it cannot scale to bigger scenarios. A suitable solution should take into account encoding enough information without increasing the number of bits added to the packets. Usually, this issue rises with SFC approaches relying on multiple encapsulation schemes.

Therefore, a tradeoff between allocating enough size for SFC information and reducing the overall packet size is needed. Also, regarding the needs for fine granularity in flow classification and customized user/application services, the scalability issues raised by granularity should be considered. Since it multiplies the flow state in networking devices, the effect of fine granularity can be dramatic when an infrastructure scales out. The switches might have exploding demand on processing resources and forwarding state. Moreover, as seen in the benchmarking presented in Section V-F, the majority of studied approaches are assessed in testbeds or using simulations based on real traffic. Thus, a performance assessment of SFC in realistic and large-scale scenarios is challenging. On the one hand, the majority of the studied approaches are mainly prototypes and not yet production ready. On the other hand, the popular header-based approaches (e.g., NSH [42]) require support and need time to be fully supported by networking devices.

3) Management: Operational and management challenges for service chaining are mainly related to high level SFC applications and infrastructure management flexibility. Further research is needed concerning the SFC management layer including SFC management applications, algorithms for smart mapping between SFs, visualization of SFs status and maintenance tools. Managing SFC operations in large-scale networks remains a hard task; a centralized SDN control and NFV orchestration permits a wide view of the network. Decomposing the SFC domain into smaller domains that can be managed locally, while interacting with other domains, simplifies the overall management of SFC. The hierarchical SFC [109] is a new architectural concept of decomposing the SFC domains, yet very few proposals [110] are published in this topic.

While SDN and NFV play an important role in simplifying SFC infrastructure management; yet the MBoxes deployment issues directly affect the SFC. The management of heterogeneous SFs in the infrastructure is challenging, as there are diverse service appliances, namely hardware appliances, virtualized appliances, open source appliances, and vendor specific appliances. The openness to the different service functions and adaptation between traditional and SDN/ NFV networks are admittedly problematic.

Some SFC solutions require specific configurations in the SFs or need special protocol support (e.g., VLAN support and interfaces configuration), hence the per-instance modification is a tedious task in large-scale infrastructures. Moreover, the support of multiple modifications of networking devices, controllers, VNFs and southbound protocols to implement an SFC solution is a challenge for the SFC deployment in production networks. Thus, further research is needed to reduce deployment complexities due to these modifications.

4) Security: Security is the concern of every network operator, from the enterprise network to carrier networks. The infrastructure security can be threatened by some security concerns caused by SFC approaches while the SFC operations can fail due to some security policies [6], [111], [112]. Usually, SFC traffic steering techniques rely on flow identification information and metadata sharing. Thus, a risk of confidentiality and integrity is highly present. The SFC information might reveal infrastructure, network, or subscribers' information. Network traffic can be intercepted, modified, or manipulated. Moreover, some MBoxes are opaque: they modify packets headers. Such modifications can cause misleading policies and lead to inconsistencies in the network. Some SFC approaches are limited to transparent SFs, where in reality an infrastructure is hybrid and contains transparent and opaque SFs.

SFs can fail due to several reasons (e.g., overload, security attack, misconfiguration, and resource unavailability). SFs failure can cause unavailability of the service chain, because the traffic intended to return from an SF is dropped for some reason and does not reach the next SFs in the service chain. Effectively, there are several SFC approaches proposed in the literature. However, the process of deployment in production environments is yet delayed. Most of the proposed approaches are prototypes and are under discussion. In practice, security MBoxes might block or drop packets carrying SFC headers or tags in transit. Some verification processes might drop SFC traffic, considering it as unauthorized or non-legitimate (e.g., verification of modified Mac addresses).

VII. Conclusion

SFC is an active research area and several schemes have been proposed in the recent literature, motivated by the support of next generation technologies such as SDN, NFV, and Cloud Computing. Different traffic steering methods are used in various SFC approaches. The SDN-based SFC approaches, analyzed in this article, were classified based on the traffic steering methods used and that is due to their impact on the performance and efficiency of SFC approaches. The studied SFC approaches were also evaluated based on the deployment cost, scalability, flexibility, and incurred overhead.

We can classify the studied traffic steering methods into three categories. The first is based on programmable switches and seems to be the most beneficial. Programmable switches-based SFC schemes are based on flow matching in the switches. Although they ensure flexibility and avoid using encapsulation and complex modifications to SFs, they present a major scalability limitation, due to the increased memory requirements of the limited switches memory. The second category of SFC approaches consists of coupling switches with compact packet tags and is less beneficial. This is due to the increase in overhead and the need for tag support at the SFC elements. Finally, the third category groups SFC approaches that are based on packet header, and represents the most complex and the least scalable method.

To conclude, the innovation in SFC started with traffic engineering and has its continuity thanks to the support of SDN, NFV, and Cloud Computing. Furthermore, a vast research area in SDN-NFV-Cloud-based traffic steering for service chaining is open, especially in optimizing delivery time and the overall quality of service, as well as the scalability of SFC solutions.

REFERENCES

[1] J. Sherry, S. Ratnasamy, and J. S. At, "A survey of enterprise middlebox deployments," EECS Dept., Univ. California at Berkeley, Berkeley, CA, USA, Rep. UCB/EECS-2012-24, 2012.

[2] J. Halpern and C. Pignataro, "Service function chaining (SFC) architecture," IETF, Fremont, CA, USA, RFC 7665, 2015. [Online]. Available: https://tools.ietf.org/html/rfc7665

[3] *SDN Architecture Overview*, Open Netw. Found., Palo Alto, CA, USA, 2013.

[4] *NFV-Architectural Framework*, ETSI, Sophia Antipolis, France, 2013. [Online]. Available: http://www.etsi.org/deliver/etsi_gs/nfv/001_099/002/01.01.01_60/gs_nfv002v010101p.pdf

[5] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing & softwarization: A survey on principles, enabling technologies & solutions," *IEEE Commun. Surveys Tuts.*, to be published.

[6] I. Farris, T. Taleb, Y. Khettab, and J. S. Song, "A survey on emerging SDN and NFV security mechanisms for IoT systems," *IEEE Commun. Surveys Tuts.*, to be published.

[7] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for Internet of Things realization," *IEEE Commun. Surveys Tuts.*, to be published.

[8] W. John *et al.*, "Research directions in network service chaining," in *Proc. Workshop Softw. Defined Netw. Future Netw. Services (SDN4FNS)*, 2013, pp. 1–7.

[9] B. Yi, X. Wang, K. Li, S. K. Das, and M. Huang, "A comprehensive survey of network function virtualization," *Comput. Netw.*, vol. 133, pp. 212–262, Mar. 2018.

[10] P. Quinn and T. Nadeau, "Problem statement for service function chaining," IETF, Fremont, CA, USA, RFC 7498, 2015. [Online]. Available: https://tools.ietf.org/html/rfc7498

[11] H.-J. Ku, J.-H. Jung, and G.-I. Kwon, "A study on reinforcement learning based SFC path selection in SDN/NFV," *Int. J. Appl. Eng. Res.*, vol. 12, no. 12, pp. 3439–3443, 2017.

[12] G. Sallam, G. R. Gupta, B. Li, and B. Ji, "Shortest path and maximum flow problems under service function chaining constraints," *arXiv Preprint arXiv1801.05795*, 2018.

[13] M. M. Tajiki, S. Salsano, M. Shojafar, L. Chiaraviglio, and B. Akbari, "Energy-efficient path allocation heuristic for service function chaining," in *Proc. 21st Conf. Innov. Clouds Internet Netw. (ICIN)*, Paris, France, 2018, pp. 20–22.

[14] N. Huin, B. Jaumard, and F. Giroire, "Optimization of network service chain provisioning," in *Proc. IEEE Int. Conf. Commun.*, Paris, France, 2017, pp. 1–7.

[15] S. Sahhaf *et al.*, "Network service chaining with optimized network function embedding supporting service decompositions," *Comput. Netw.*, vol. 93, pp. 492–505, Dec. 2015.

[16] P. Wang, J. Lan, X. Zhang, Y. Hu, and S. Chen, "Dynamic function composition for network service chain: Model and optimization," *Comput. Netw.*, vol. 92, pp. 408–418, Dec. 2015.

[17] S. Sahhaf, W. Tavernier, D. Colle, and M. Pickavet, "Network service chaining with efficient network function mapping based on service decompositions," in *Proc. 1st IEEE Conf. Netw. Softwarization (NetSoft)*, London, U.K., 2015, pp. 1–5.

[18] A. M. Medhat, G. Carella, C. Lück, M.-I. Corici, and T. Magedanz, "Near optimal service function path instantiation in a multi-datacenter environment," in *Proc. 11th Int. Conf. Netw. Service Manag. (CNSM)*, Barcelona, Spain, 2015, pp. 336–341.

[19] A. Gushchin, A. Walid, and A. Tang, "Enabling service function chaining through routing optimization in software defined networks," in *Proc. 53rd Annu. Allerton Conf. Commun. Control Comput. (Allerton)*, Monticello, IL, USA, 2016, pp. 573–581.

[20] H. Hantouti and N. Benamar, "A novel SDN based architecture and traffic steering method for service function chaining," in *Proc. Int. Conf. Sel. Topics Mobile Wireless Netw.*, Tangier, Morocco, 2018.

[21] O. Soualah, M. Mechtri, C. Ghribi, and D. Zeghlache, "An efficient algorithm for virtual network function placement and chaining," in *Proc. 14th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Las Vegas, NV, USA, 2017, pp. 647–652.

[22] Z. Allybokus, N. Perrot, J. Leguay, L. Maggi, and E. Gourdin, "Virtual function placement for service chaining with partial orders and anti-affinity rules," *Networks*, vol. 71, no. 2, pp. 97–106, 2017.

[23] D. Bhamare *et al.*, "Optimal virtual network function placement in multi-cloud service function chaining architecture," *Comput. Commun.*, vol. 102, pp. 1–16, Apr. 2017.

[24] M. C. Luizelli, W. L. da Costa Cordeiro, L. S. Buriol, and L. P. Gaspary, "A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining," *Comput. Commun.*, vol. 102, pp. 67–77, Apr. 2017.

[25] Y. Sang, B. Ji, G. R. Gupta, X. Du, and L. Ye, "Provably efficient algorithms for joint placement and allocation of virtual network functions," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Atlanta, GA, USA, 2017, pp. 1–9.

[26] C. Pham, N. H. Tran, S. Ren, W. Saad, and C. S. Hong, "Traffic-aware and energy-efficient VNF placement for service chaining: Joint sampling and matching approach," *IEEE Trans. Services Comput.*, to be published.

[27] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *Proc. IEEE 3rd Int. Conf. Cloud Netw. (CloudNet)*, 2014, pp. 7–13.

[28] M. Mechtri, C. Ghribi, and D. Zeghlache, "A scalable algorithm for the placement of service function chains," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 13, pp. 533–546, Sep. 2016.

[29] M. Ghaznavi, N. Shahriar, R. Ahmed, and R. Boutaba, "Service function chaining simplified," *arXiv Preprint arXiv1601.00751*, 2016.

[30] C. Ghribi, M. Mechtri, O. Soualah, and D. Zeghlache, "SFC provisioning over NFV enabled clouds," in *Proc. IEEE 10th Int. Conf. Cloud Comput. (CLOUD)*, 2017, pp. 423–430.

[31] T. Taleb, A. Ksentini, M. Chen, and R. Jantti, "Coping with emerging mobile social media applications through dynamic service function chaining," *IEEE Trans. Wireless Commun.*, vol. 15, no. 4, pp. 2859–2871, Apr. 2016.

[32] R. A. Eichelberger, T. Ferreto, S. Tandel, and P. A. P. R. Duarte, "SFC path tracer: A troubleshooting tool for service function chaining," in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manag. (IM)*, Lisbon, Portugal, 2017, pp. 568–571.

[33] L. Bondan, T. Wauters, B. Volckaert, F. D. Turck, and L. Z. Granville, "Anomaly detection framework for SFC integrity in NFV environments," in *Proc. IEEE Conf. Netw. Softwarization (NetSoft)*, Bologna, Italy, 2017, pp. 1–5.

[34] W. Lee and N. Kim, "Security policy scheme for an efficient security architecture in software-defined networking," *Information*, vol. 8, no. 2, p. 65, 2017.

[35] K. Fysarakis *et al.*, "A reactive security framework for operational wind parks using service function chaining," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, 2017, pp. 663–668.

[36] G. Li *et al.*, "Fuzzy theory based security service chaining for sustainable mobile-edge computing," *Mobile Inf. Syst.*, vol. 2017, p. 13, Feb. 2017.

[37] H. Wang *et al.*, "SICS: Secure in-cloud service function chaining," *arXiv Preprint arXiv1606.07079*, vol. abs/1606.0, 2016.

[38] S. Hyun *et al.*, "Interface to network security functions for cloud-based security services," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 171–178, Jan. 2018.

[39] D. Bhamare, R. Jain, M. Samaka, and A. Erbad, "A survey on service function chaining," *J. Netw. Comput. Appl.*, vol. 75, pp. 138–155, Nov. 2016.

[40] Y. Xie, Z. Liu, S. Wang, and Y. Wang, "Service function chaining resource allocation: A survey," *arXiv Preprint arXiv1608.00095*, 2016.

[41] A. M. Medhat *et al.*, "Service function chaining in next generation networks: State of the art and research challenges," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 216–223, Feb. 2017.

[42] P. Quinn, "Network service header," IETF-Draft, 2017. [Online]. Available: https://tools.ietf.org/html/draft-ietf-sfc-nsh-28

[43] N. McKeown *et al.*, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.

[44] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman. (2011). *Network Configuration Protocol (NETCONF)*. [Online]. Available: https://tools.ietf.org/pdf/rfc6241.pdf

[45] *Industry Specification Groups (ISGs)*, ETSI, Sophia Antipolis, France. [Online]. Available: http://www.etsi.org/about/how-we-work/how-we-organize-our-work/industry-specification-groups-isgs

[46] *European Telecommunications Standards Institute*. Accessed: Jun. 10, 2017. [Online]. Available: http://www.etsi.org/about/what-we-are

[47] *Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV, V1.1.1*, ETSI Standard GS NFV 003, 2013. [Online]. Available: http://www.etsi.org/deliver/etsi_gs/NFV/001_099/003/01.02.01_60/gs_NFV003v010201p.pdf

[48] M. S. Bonfim, K. L. Dias, and S. F. L. Fernandes, "Integrated NFV/SDN architectures: A systematic literature review," *arXiv Preprint arXiv1801.01516*, 2018.

[49] Y. Li and M. Chen, "Software-defined network function virtualization: A survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015.

[50] S. Sezer *et al.*, "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Commun. Mag.*, vol. 51, no. 7, pp. 36–43, Jul. 2013.

[51] M. Chiosi *et al.*, *Network Functions Virtualisation—Introductory White Paper Network*, ETSI, Sophia Antipolis, France, pp. 1–16, 2012.

[52] W. Simpson, "IP in IP tunneling," IETF, Fremont, CA, USA, RFC 1853, 1995.

[53] M. Mahalingam *et al.*, "VXLAN: A framework for overlaying virtualized layer 2 networks over layer 3 networks," IETF, Fremont, CA, USA, RFC 7348, pp. 1–23, Oct. 2014.

[54] P. Quinn, P. Maino, M. Smith, and X. Xu, "Generic protocol extension for VXLAN," IETF, Fremont, CA, USA, IETF-Draft, pp. 1–22, 2015.

[55] D. Meyer, "Generic routing encapsulation (GRE) status," IETF, Fremont, CA, USA, IETF-Draft, pp. 1–9, 2000.

[56] "Service function chaining (SFC)," IETF, Fremont, CA, USA, Working Group. Accessed: Dec. 8, 2015. [Online]. Available: https://datatracker.ietf.org/wg/sfc/documents/

[57] E. Bash and H. Zhang, "Service chain header," IETF, Fremont, CA, USA, Internet Draft, 2014. [Online]. Available: https://tools.ietf.org/html/draft-zhang-sfc-sch-00

[58] M. Boucadair and C. Jacquenet, "An IPv6 extension header for service function chaining (SFC)," IETF, Fremont, CA, USA, Internet Draft, 2016. [Online]. Available: https://tools.ietf.org/html/draft-jacquenet-sfc-ipv6-eh-00

[59] V. Mehmeri *et al.*, "Optical network as a service for service function chaining across datacenters," in *Proc. Opt. Fiber Commun. Conf. Exhibit. (OFC)*, Los Angeles, CA, USA, 2017, pp. 1–3.

[60] S. Kulkarni, M. Arumaithurai, K. K. Ramakrishnan, and X. Fu, "NeoNSH: Towards scalable and efficient dynamic service function chaining of elastic network functions," in *Proc. IEEE 20th Conf. Innov. Clouds Internet Netw. (ICIN)*, Paris, France, 2017, pp. 308–312.

[61] A. Conta and S. Deering, "Generic packet tunneling in IPv6 specification," IETF, Fremont, CA, USA, RFC 2473, 1998.

[62] E. S. Previdi and C. Filsfils, "IPv6 segment routing header (SRH)," IETF, Fremont, CA, USA, Internet Draft, 2016. [Online]. Available: https://tools.ietf.org/html/draft-ietf-6man-segment-routing-header-02

[63] A. Abdelsalam *et al.*, "Implementation of virtual network function chaining through segment routing in a Linux-based NFV infrastructure," in *Proc. IEEE Conf. Netw. Softwarization*, Bologna, Italy, 2017, pp. 1–5.

[64] T. Li, H. Zhou, and H. Luo, "A new method for providing network services: Service function chain," *Opt. Switch. Netw.*, vol. 26, pp. 60–68, Nov. 2017.

[65] J. Blendin, J. Ruckert, N. Leymann, G. Schyguda, and D. Hausheer, "Position paper: Software-defined network service chaining," in *Proc. 3rd Eur. Workshop Softw. Defined Netw. (EWSDN)*, London, U.K., 2014, pp. 109–114.

[66] W. Ding, W. Qi, J. Wang, and B. Chen, "OpenSCaaS: An open service chain as a service platform toward the integration of SDN and NFV," *IEEE Netw.*, vol. 29, no. 3, pp. 30–35, May/Jun. 2015.

[67] Z. A. Qazi *et al.*, "SIMPLE-fying middlebox policy enforcement using SDN," in *Proc. ACM SIGCOMM Conf. (SIGCOMM)*, Hong Kong, 2013, pp. 27–38.

[68] S. K. Fayazbakhsh, L. Chiang, V. Sekar, M. Yu, and J. C. Mogul, "Enforcing network-wide policies in the presence of dynamic middlebox actions using flowtags," in *Proc. 11th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, Seattle, WA, USA, 2014, pp. 543–546.

[69] B. Martini *et al.*, "SDN controller for context-aware data delivery in dynamic service chaining," in *Proc. 1st IEEE Conf. Netw. Softwarization Softw. Defined Infrastruct. Netw. Clouds IoT Services (NETSOFT)*, London, U.K., Jun. 2015, pp. 1–5.

[70] P. B. Pawar and K. Kataoka, "Segmented proactive flow rule injection for service chaining using SDN," in *Proc. IEEE NetSoft Conf. Workshops Softw. Defined Infrastruct. Netw. Clouds IoT Services*, Seoul, South Korea, 2016, pp. 38–42.

[71] A. Abujoda, H. R. Kouchaksaraei, and P. Papadimitriou, "SDN-based source routing for scalable service chaining in datacenters," in *Proc. Int. Conf. Wired/Wireless Internet Commun.*, Thessaloniki, Greece, 2016, pp. 66–77.

[72] I. Trajkovska *et al.*, "SDN-based service function chaining mechanism and service prototype implementation in NFV scenario," *Comput. Stand. Interfaces*, vol. 54, pp. 247–265, Nov. 2017.

[73] "Source packet routing in networking (spring)," IETF, Fremont, CA, USA, Working Group. Accessed: Dec. 12, 2016. [Online]. Available: https://datatracker.ietf.org/wg/spring/about/

[74] Z. Li and X. Xu, "Service function chaining using MPLS-SPRING," IETF, Fremont, CA, USA, Internet Draft, 2016. [Online]. Available: https://tools.ietf.org/html/draft-xu-sfc-using-mpls-spring-03.html

[75] I. Cerrato *et al.*, "User-specific network service functions in an SDN-enabled network node," in *Proc. 3rd Eur. Work. Softw. Defined Netw. (EWSDN)*, London, U.K., 2014, pp. 135–136.

[76] A. Bianco, R. Birke, L. Giraudo, and M. Palacin, "OpenFlow switching: Data plane performance," in *Proc. IEEE Int. Conf. Commun.*, 2010, pp. 1–5.

[77] X. Wang, Q. Zhang, I. Kim, P. Palacharla, and M. Sekiya, "Object-oriented network virtualization—An evolution from SDN to SPN," in *Proc. Adv. Photon. Commun.*, 2014, p. NT1C.2.

[78] OASIS TOSCA. *TOSCA Simple Profile for Network Functions Virtualization (NFV) Version 1.0*. Accessed: May 11, 2017. [Online]. Available: http://docs.oasis-open.org/tosca/tosca-nfv/v1.0/tosca-nfv-v1.0.html

[79] OpenStack. *OpenStack*. Accessed: Apr. 22, 2017. [Online]. Available: https://www.openstack.org/

[80] NoxRepo and Contributors. *The Pox Controller*. Accessed: Mar. 4, 2017. [Online]. Available: https://github.com/noxrepo/pox

[81] F. Paolucci, F. Cugini, A. Giorgetti, N. Sambo, and P. Castoldi, "A survey on the path computation element (PCE) architecture," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 4, pp. 1819–1841, 4th Quart., 2013.

[82] Y. Zhang *et al.*, "StEERING: A software-defined networking for inline service chaining," in *Proc. 21st IEEE Int. Conf. Netw. Protocols (ICNP)*, Goettingen, Germany, 2013, pp. 1–10.

[83] T-Nova. (2014). *T-Nova Project*. Accessed: Mar. 15, 2017. [Online]. Available: http://www.t-nova.eu/

[84] *The OpenDaylight Platform*, OpenDaylight, San Francisco, CA, USA. Accessed: Feb. 11, 2016. [Online]. Available: https://www.opendaylight.org/

[85] Openstack. *OpenStack User's Survey 2016*. Accessed: Oct. 2, 2016. [Online]. Available: https://www.openstack.org/assets/survey/April-2016-User-Survey-Report.pdf

[86] ICCLab-ZHAW. *Netfloc SDK4SDN*. Accessed: May 23, 2017. [Online]. Available: http://icclab.github.io/netfloc/

[87] OPNFV. *Yardstick Project OPNFV*. Accessed: May 16, 2017. [Online]. Available: https://wiki.opnfv.org/display/yardstick/Yardstick

[88] A. Csoma *et al.*, "ESCAPE: Extensible service chain prototyping environment using mininet, click, NETCONF and POX," in *Proc. ACM Conf. SIGCOMM Demo*, Chicago, IL, USA, 2014, pp. 125–126.

[89] OpenDayLight. *Service Function Chaining*. Accessed: Aug. 17, 2016. [Online]. Available: https://wiki.opendaylight.org/view/Service_Function_Chaining:Main

[90] A. Rodriguez-Natal *et al.*, "LISP: A southbound SDN protocol?" *IEEE Commun. Mag.*, vol. 53, no. 7, pp. 201–207, Jul. 2015.

[91] J. G. Herrera and J. F. Botero, "Resource allocation in NFV: A comprehensive survey," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 518–532, Sep. 2016.

[92] Y. Li, F. Zheng, M. Chen, and D. Jin, "A unified control and optimization framework for dynamical service chaining in software-defined NFV system," *IEEE Wireless Commun.*, vol. 22, no. 6, pp. 15–23, Dec. 2015.

[93] E. Hernandez-Valencia, S. Izzo, and B. Polonsky, "How will NFV/SDN transform service provider OPEX?" *IEEE Netw.*, vol. 29, no. 3, pp. 60–67, May/Jun. 2015.

[94] M. T. Beck and J. F. Botero, "Scalable and coordinated allocation of service function chains," *Comput. Commun.*, vol. 102, pp. 78–88, Apr. 2017.

[95] M. Boucadair and C. Jaquenet, "Service function chaining: Design considerations, analysis & recommendations," IETF-Draft, 2014. [Online]. Available: https://tools.ietf.org/html/draft-boucadair-sfc-design-analysis-03

[96] A. Ksentini, M. Bagaa, and T. Taleb, "On using SDN in 5G: The controller placement problem," in *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, Washington, DC, USA, 2016, pp. 1–6.

[97] A. Ksentini, M. Bagaa, T. Taleb, and I. Balasingham, "On using bargaining game for optimal placement of SDN controllers," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kuala Lumpur, Malaysia, 2016, pp. 1–6.

[98] M. Bagaa, T. Taleb, A. Laghrissi, A. Ksentini, and H. Flinck, "Coalitional game for the creation of efficient virtual core network slices in 5G mobile systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 469–484, Mar. 2018.

[99] A. Laghrissi, T. Taleb, and M. Bagaa, "Conformal mapping for optimal network slice planning based on canonical domains," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 519–528, Mar. 2018.

[100] T. Taleb, M. Bagaa, and A. Ksentini, "User mobility-aware virtual network function placement for virtual 5G network infrastructure," in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, U.K., 2015, pp. 3879–3884.
[101] M. Bagaa, T. Taleb, and A. Ksentini, "Service-aware network function placement for efficient traffic handling in carrier cloud," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Istanbul, Turkey, 2014, pp. 2402–2407.
[102] T. Taleb, B. Mada, M.-I. Corici, A. Nakao, and H. Flinck, "PERMIT: Network slicing for personalized 5G mobile telecommunications," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 88–93, May 2017.
[103] T. Taleb, A. Ksentini, and B. Sericola, "On service resilience in cloud-native 5G mobile systems," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 483–496, Mar. 2016.
[104] H. Hantouti, N. Benamar, and T. Taleb, "A novel compact header for traffic steering in service function chaining," in *Proc. IEEE Int. Conf. Commun.*, Kansas City, MO, USA, 2018, pp. 1–6.
[105] *Multi-Access Edge Computing*, ETSI, Sophia Antipolis, France. Accessed: Jun. 30, 2018. [Online]. Available: http://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing
[106] T. Taleb *et al.*, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.
[107] B. Xiong, K. Yang, J. Zhao, W. Li, and K. Li, "Performance evaluation of OpenFlow-based software-defined networks based on queueing model," *Comput. Netw.*, vol. 102, pp. 172–185, Jun. 2016.
[108] F. Dürr and T. Kohler, "Comparing the forwarding latency of OpenFlow hardware and software switches," Fak. Inform. Elektrotechnik und Informationstechnik, Univ. Stuttgart, Stuttgart, Germany, Rep. TR 2014/04, p. 18, 2014.
[109] D. Dolson, S. Homma, D. Lopez, and M. Boucadair, "Hierarchical service function chaining (hSFC)," IETF, Fremont, CA, USA, Internet Draft, 2018. [Online]. Available: https://tools.ietf.org/pdf/draft-ietf-sfc-hierarchical-07.pdf
[110] A.-V. Vu and Y. Kim, "An implementation of hierarchical service function chaining using OpenDaylight platform," in *Proc. IEEE NetSoft Conf. Workshops (NetSoft)*, Seoul, South Korea, 2016, pp. 411–416.
[111] Y. Khettab, M. Bagaa, D. L. C. Dutra, T. Taleb, and N. Toumi, "Virtual security as a service for 5G verticals," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Barcelona, Spain, 2018, pp. 1–6.
[112] S. Lal, T. Taleb, and A. Dutta, "NFV: Security threats and best practices," *IEEE Commun. Mag.*, vol. 55, no. 8, pp. 211–217, May 2017.
[113] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with rocketfuel," in *Proc. Conf. Appl. Technol. Archit. Protocols Comput. Commun. (SIGCOMM)*, Pittsburgh, PA, USA, 2002, pp. 133–145.
[114] C. Xie, W. Meng, and C. Wang, and B. Khasnabish, "Service function chain use cases in broadband," IETF, Fremont, CA, USA, Internet Draft, 2015. [Online]. Available: https://tools.ietf.org/html/draft-meng-sfc-broadband-usecases-04

Hajar Hantouti received the M.S. degree in information systems security from the School of Applied Sciences, University of Ibn Tofail, Morocco, in 2014, with a dissertation on memory and computer forensics. She is currently pursuing the Ph.D. degree with the School of Science, University of Moulay Ismail, Morocco. Her research interests include service function chaining, software defined networking, and network function virtualization.

Nabil Benamar received the B.E., M.Sc., and Ph.D. degrees from the University of Moulay Ismail in 1998, 2001, and 2004, respectively. He is currently a Professor of computer sciences with the School of Technology, Department of Computer Engineering, Moulay Ismail University, Morocco. His main research topics are IPv6, vehicular networks, ITS, IoT, and Service Function Chaining. He has authored several journal papers and IETF Internet Drafts. He is a reviewer for *Computer Communications* (Elsevier), JKSUCS (Elsevier), Adhoc (Elsevier), IJWIN (Springer), AJSE (Springer), and IEEE ACCESS. He is also a TPC Member in different IEEE conferences, such as Globecom, ICC, and PIMRC. He is an IPv6 Expert (he.net certified) and a Consultant with many international organisms, such as Academic Agency for Francophonie, AFRINIC, and MENOG. He is also an expert in Internet Governance after completion the ISOC Next generation e-learning programme: "Shaping the Internet, History and Futures" in 2012. He is an ISOC Ambassador to IGF from 2012 to 2013, a Google Panelist in the first Arab-IGF, an ISOC Fellow to IETF'89&92&95&99 and ICANN'50&54 Fellow. He is a member of Task Force for Arabic IDNs which is a part of Global Stakeholder Engagement endorsed by ICANN. He is a member of G6 association for IPv6 and one of the contributors to the IPv6 MOOC.

Tarik Taleb received the B.E. degree (with Distinction) in information engineering, and the M.Sc. and Ph.D. degrees in information sciences from Tohoku University in 2001, 2003, and 2005, respectively. He is currently a Professor with the School of Electrical Engineering, Department of Communications and Networking, Aalto University, Espoo, Finland. He was a Senior Researcher and 3GPP Standards Expert with NEC Europe Ltd. Until 2009, he was an Assistant Professor with the Graduate School of Information Sciences, Tohoku University, Japan, in a lab fully funded by KDDI. His research interests lie in the field of architectural enhancements to mobile core networks (particularly 3GPP's), mobile cloud networking, mobile multimedia streaming, and social media networking. He has also been directly engaged in the development and standardization of the Evolved Packet System. He was a recipient of many awards for his many contributions in the area of mobile networking. He is a member of the IEEE Communications Society Standardization Program Development Board and serves as the Steering Committee Chair of the IEEE Conference on Standards for Communications and Networking.

Abdelquoddouss Laghrissi received the bachelor's degree in mathematics and computer science and the master's degree in applied computer science with a dissertation on empathy in vehicular ad hoc networks from the School of Sciences, Mohamed V. University in 2012 and 2014, respectively. He is currently pursuing the Ph.D. degree with the School of Electrical Engineering, Aalto University, Finland. From 2014 to 2015, he was a Volunteer Member of the cloud computing working group within a Euro-Mediterranean project MOSAIC on Cooperation with Mediterranean Partners to build Opportunities around ICT and Societal and Industrial Challenges of H2020, and is currently involved in a European project 5G!Pagoda on network slicing and 5G in the context of H2020, during which he published articles and contributed to several deliverables. He is a member of the MOSA!C Laboratory. His research interests include mobile cloud computing, network function virtualization, and software defined networking.