Computer Networks 133 (2018) 212–262

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Review article

A comprehensive survey of Network Function Virtualization

Bo Yi[a], Xingwei Wang[b,*], Keqin Li[c], Sajal k. Das[d], Min Huang[e]

[a] College of Computer Science and Engineering, Northeastern University, Shenyang 110169, China
[b] College of Software, Northeastern University, Shenyang 110169, China
[c] Department of Computer Science, State University of New York, New York 12561, USA
[d] Department of Computer Science, Missouri University of Science and Technology, Rolla, MO 65409, USA
[e] College of Information Science and Engineering, State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China

ARTICLE INFO

Article history:
Received 13 April 2017
Revised 15 January 2018
Accepted 17 January 2018
Available online 31 January 2018

Keywords:
Network Function Virtualization
Virtual network function
Software defined networking
Algorithm
Network softwarization

ABSTRACT

Today's networks are filled with a massive and ever-growing variety of network functions that coupled with proprietary devices, which leads to network ossification and difficulty in network management and service provision. Network Function Virtualization (NFV) is a promising paradigm to change such situation by decoupling network functions from the underlying dedicated hardware and realizing them in the form of software, which are referred to as Virtual Network Functions (VNFs). Such decoupling introduces many benefits which include reduction of Capital Expenditure (CAPEX) and Operation Expense (OPEX), improved flexibility of service provision, etc. In this paper, we intend to present a comprehensive survey on NFV, which starts from the introduction of NFV motivations. Then, we explain the main concepts of NFV in terms of terminology, standardization and history, and how NFV differs from traditional middlebox based network. After that, the standard NFV architecture is introduced using a bottom up approach, based on which the corresponding use cases and solutions are also illustrated. In addition, due to the decoupling of network functionalities and hardware, people's attention is gradually shifted to the VNFs. Next, we provide an extensive and in-depth discussion on state-of-the-art VNF algorithms including VNF placement, scheduling, migration, chaining and multicast. Finally, to accelerate the NFV deployment and avoid pitfalls as far as possible, we survey the challenges faced by NFV and the trend for future directions. In particular, the challenges are discussed from bottom up, which include hardware design, VNF deployment, VNF life cycle control, service chaining, performance evaluation, policy enforcement, energy efficiency, reliability and security, and the future directions are discussed around the current trend towards network softwarization.

Contents

* Corresponding author.
E-mail addresses: yibobooscar@gmail.com (B. Yi), wangxw@mail.neu.edu.cn (X. Wang), lik@newpaltz.edu (K. Li), sdas@mst.edu (S.k. Das), mhuang@mail.neu.edu.cn (M. Huang).

https://doi.org/10.1016/j.comnet.2018.01.021
1389-1286/

1. Introduction

Currently, most traditional networks are full of various proprietary hardware appliances which are also called middle-boxes [1] such as firewall and Network Address Translator (NAT). A given service usually has a strong connection with some specific middle-boxes. For example, launching a new service needs to deploy a variety of middle-boxes and accommodating these middle-boxes is becoming more and more difficult. In addition, designing proprietary hardware based protocols and deploying proprietary hardware are extremely hard, expensive and time-consuming. One typical example is that the procedure of transforming IPv4 to IPv6 has continued for over ten years, and yet IPv4 is still used widely. Thus, it is extremely difficult to update a protocol running on proprietary hardware, and let alone deploying a new one [2] and [3]. Moreover, with the ever increasing and various service requirements, service providers have to scale up their physical infrastructure periodically, which directly leads to high CAPital EXpenditure (CAPEX) and OPeration EXpenses (OPEX) [4].

The Commercial-Off-The-Shelf (COTS) network equipment (e.g., $\times 86$ based hardware), which can satisfy the needs of general use rather than customized purposes, are providing far more capacities with less cost than specialized network equipment. Hence, the COTS hardware has come up as a highly competitive force against dedicated hardware. In this way, most Telecommunications Operators (TOs) look forward to separating network functions from the purpose-built devices and implementing them as software which could be deployed on standard COTS hardware. Under this situation, over twenty of the world's largest TOs, such as American Telephone and Telegraph (AT&T), British Telecom (BT) and Deutsche Telekom (DT), formed an Industry Specification Group (ISG) within the European Telecommunications Standards Institute (ETSI) to define Network Function Virtualization (NFV) in October 2012 [5] (the acronyms ETSI and ETSI NFV ISG are used synonymously hereafter and all the acronyms used in this work are summarized in Table 1). Due to the separation of network function from hardware, NFV can effectively reduce the CAPEX and OPEX. Since then, the ETSI has grown to a large community with 300+ members all over the world including 38 of the world's

Table 1
Acronyms used throughout this work.

Abbreviations	Full name	Abbreviations	Full name	Abbreviations	Full name
3GPP	3rd Generation Partnership Project Service	ATIS	Alliance for Telecommunications Industry Solutions	AT&T	American Telephone and Telegraph
BBF	Broadband Forum	BNG	Broadband Network Gateway	BSS	Business Support System
BT	British Telecom	CAPEX	Capital Expenditure	CG-NAT	Carrier Grade Network Address Translator
COTS	Commercial-Off-The-Shelf	CDN	Content Distribution Network	CPE	Customer Premise Equipment
C-RAN	Cloud RAN	CS/MG-CF	Call Session/Media Gateway Control Function	DAS	Direct Attached Storage
DDoS	Distributed Denial of Service	DHCP	Dynamic Host Configuration Protocol	DMTF	Distributed Management Task Force
DNS	Domain Name System	DPDK	Data Plane Development Kit	DPI	Deep Packet Inspection
DROP	Distributed Router Open Platform	DT	Deutsche Telekom	DVR	Distributed Virtual Router
EM	Element Manager	EMS	Element Management System	eNodeB	Evolved Node B
EPC	Evolved Packet Core	ETSI	European Telecommunications Standards Institute	FD.io	Fast Data I/O
FM	Flow Monitor	FMC	Fixed Mobile Convergence	GA	Genetic Algorithm
GAL	Green Abstraction Layer	GENI	Global Environment for Networking Innovation	GPRS	General Packet Radio Service
GPU	Graphics Processing Unit	G/S-GSN	Gateway/Serving GPRS Supported Node	HDD	Hard Disk Driver
HLR	Home Location Register	HSS	Home Subscriber Server	IaaS	Infrastructure as a Service
ICN	Information-Centric Networking	IDS	Intrusion Detection System	IETF	Internet Engineering Task Force
ILP	Integer Linear Programming	IMS	IP Multimedia Subsystem	IoT	Internet of Things
IPS	Intrusion Prevention System	IRTF	Internet Research Task Force	ISG	Industry Specification Group
KVM	Kernel-based Virtual Machine	LTE	Long Term Evolution	MANO	Management and Orchestration
MILP	Mixed ILP	MME	Mobility Management Entity	NAS	Network Attached Storage
NETCONF	Network Configuration Protocol	NFV	Network Function Virtualization	NFVI	NFV Infrastructure
NFVIaaS	NFVI as a Service	NFVO	NFV Orchestrator	NFVRG	NFV Research Group
NIC	Network Interface Card	N-PoP	NFVI Point of Presence	NSH	Network Service Header
NV	Network Virtualization	OEO	Optical-Electricity-Optical	ONF	Open Networking Foundation
OPEN-O	OPEN Orchestrator	OPEX	Operations Expenses	OSM	Open Source MANO
OSS	Operational Support System	OTN	Optical Transport Network	OVF	Open Virtualization Format
OVS	Open vSwitch	PaaS	Platform as a Service	PNF	Physical Network Function
RAN	Radio Access Network	RHEL	Red Hat Enterprise Linux	RNC	Radio Network Controller
SA	Simulated Annealing	SaaS	Software as a Service	SAL	Service Availability Levels
SAN	Storage Area Network	SDC	Software Defined Compute	SDN	Software-Defined Networking
SDO	Standard Development Organizations	SDS	Software Defined Storage	SEG	Security Expert Group
SFC WG	Service Function Chaining Working Group	SFP	Service Function Path	S/G-GSN	Serving/Gateway GPRS Support Node
SLA	Service Level Agreement	S/P-GW	Serving/Public data network GateWay	SR-IOV	Single Root I/O Virtualization
SSD	Solid State Disk	TaaS	Tap as a Service	TE	Traffic Engineering
TO	Telecommunications Operators	vE-CPE	Virtuallized Enterprise CPE	VIM	Virtualized Infrastructure Manager
VLAN	Virtual Local Area Network	VM	Virtual Machine	VMM	Virtual Machine Monitor
VNE	Virtual Network Embedding	VNF	Virtual Network Function	VNFaaS	VNF as a Service
VNFC	VNF Component	VNF-C	VNF Chaining	VNF FG	VNF Forwarding Graph
VNFM	VNF manager	VNF-M	VNF Migration	VNF-MC	VNF Multicast
VNF-P	VNF Placement	VNF-S	VNF Scheduling	VNPaaS	Virtual Network Platform as a Service
VPE	Virtualization Polling Engine	VPN	Virtual Private Network	ZOOM	Zero-time Orchestration, Operations and Management

major service providers. These members work intensely to develop the required standards for NFV as well as share their experience of NFV implementations and testing [6] and [7].

NFV transforms the way that TOs build network by utilizing standard IT virtualization technologies, that is, consolidating various types of proprietary network equipment onto COTS based high volume equipment [8]. Based on the current development of virtualization technologies, the appearance of NFV makes it possible for most TOs to achieve strong network flexibility and fast new service deployment cycle. In this way, TOs can satisfy the continuously growing customer requirements easily and reduce the network operation and maintenance cost at the same time. Nevertheless, challenges are always accompanying opportunities. The network flexibility is achieved by introducing the virtualization plane, which may result in many new issues such as security and scalability.

In principle, all network functions and other network elements can be considered for virtualization. These virtualized instances are referred to as Virtual Network Functions (VNFs) in the context of NFV, which provide the same functionalities as the corresponding physical instances. Besides, VNFs can be instantiated, executed and deployed by service providers in the NFV Infrastructure (NFVI) environment which provides the required resources (e.g., compute and storage). Typically, chaining multiple VNFs in a particular order can constitute a specific service. To provide the VNF constituted services, most enterprises play more like service consumers, because they can use resources in a pay-per-use manner instead of purchasing, configuring, and deploying the infrastructure. In addition, NFV enables service providers with certain extent flexibility, such that they can adjust the resources allocated to VNFs to satisfy the dynamically changing workload of VNFs. This mechanism promotes network resource utilization and the agility of network service provision [9].

Currently, there are many literatures researching NFV and we categorize them into three kinds according to their focus. The first kind of literatures focused on the integration of NFV and other

paradigms. For example, Munoz et al. [10] and Nejabati et al. [11] applied NFV into the optical networks, Omnes et al. [12] applied NFV into the Internet of Things (IoT) and Akyildiz et al. [13] Yang et al. [14] and Hawilo et al. [15] applied NFV into the 5G supported networks. Although these literatures explored many potential of NFV, they directly adopted the standard NFV structure without any modification. However, the standard NFV structure was proposed as a generic concept. In other words, the standard NFV structure was not specially designed for one specific network scenario and the direct usage of it might lead to many unexpected issues which were not discussed in the above works. Besides, many works also targeted on integrating NFV with Software Defined Networking (SDN) [16–19], cloud computing [20] and [21], etc. Due to the high complementary features between SDN and NFV, their integration has been widely recognized. However, not all the integrations with NFV are easy. For example, the integration between NFV supported systems and traditional systems may suffer from many compatibility issues. Despite the fact that some technologies in cloud can be used to accelerate the NFV evolvement, the design of NFVI should be prevented from falling into the design of Infrastructure as a Service (IaaS) model of cloud. The second kind of literatures focused on algorithms of many hot topics in NFV, for example, the VNF placement, scheduling and migration. All the existing algorithms can be classified as either exact ones or heuristic ones. It is known that the exact algorithms offer optimal solutions which are largely limited by the network scale [22]. Although the solutions offered by heuristic algorithms are not optimal, they are not limited by the network scale [23]. Due to the inherent features of exact algorithms, more and more heuristic algorithms are designed to offer near-optimal solutions with small execution time.

The last kind of literatures focused on NFV surveys and reviews (e.g., Refs. [24–28]). However, some of them only introduced specific aspects of NFV. For example, Han et al. [25], Mijumbi et al. [26] and Contreras et al. [28] surveyed NFV challenges in terms of innovation, management and operation respectively, while they failed to present a comprehensive review of NFV to the persons who were unfamiliar to it. In addition, some works intended to present a survey of NFV in other scenarios, for example, NFV in 5G [27]. Unfortunately, Abdelwahab et al. [27] focused more on 5G instead of NFV. It is true that some works indeed presented a relatively comprehensive survey on NFV (e.g., [24]). Mijumbi et al. [24] not only explained the basic knowledge of NFV, but also compared NFV with other popular concepts to highlight the business model of NFV. However, since the attention is shifted from hardware to software, the key algorithmic aspects of NFV hot topics are not discussed in Ref. [24]. Therefore, a diverse and comprehensive survey of NFV is still desired.

In this work, we not only present a complete and detailed overview of NFV, but also summarize the popular VNF related algorithms in terms of VNF placement, scheduling, migration, chaining and multicast. Besides, the challenges for NFV are discussed in a bottom up manner and the future research directions and application scenarios of NFV are also discussed. The main contributions of this work are summarized as follows:

- Considering the important role that NFV may play in the future, we summarize the existing works with respect to the motivation, terminologies, standardization activities, history, architecture, NFV use cases and solutions in order to provide a comprehensive and detailed introduction for researchers who are new to NFV. In particular, the NFV architecture is presented in a bottom up manner, which includes the physical infrastructure, virtualization layer, virtual infrastructure, management and orchestration layer, and VNF layer.
- Decoupling network functions from proprietary hardware and implementing them as VNFs result in a situation that more and more efforts are contributed to the design and implementation of VNF related algorithms. Apparently, VNF algorithms play an important role in the evolvement of NFV. However, to the best of our knowledge, there is no tutorial and review work about VNF algorithms. Therefore, in this paper, we present an extensive and in-depth discussion of the VNF related algorithms in terms of many popular aspects which include VNF placement, scheduling, migration, chaining and multicast.
- Although a lot of experiences accumulated during the NFV evolvement and deployment, we should be aware that there are still many obstacles required to be overcome before using NFV in production networks. Thus, to help address these obstacles and avoid pitfalls as far as possible, we first discuss the major challenges that NFV might face, and introduce the related experience that might be used to address such challenges. In particular, these challenges are illustrated from bottom up, which include hardware design, VNF deployment, VNF life cycle control, service chaining, performance evaluation, policy enforcement, energy efficiency, reliability and security.
- NFV can implement its goals independently. However, the combination with other popular concepts (e.g., SDN) would bring significant benefits. Currently, the integration of NFV and SDN is leading a trend towards network softwarization. Thus, we first discuss the future directions of NFV in terms of software defined infrastructure, control and application respectively. Then, with the advent of new paradigms such as 5G and Internet of Things (IoT), network softwarization is attracting more and more attention. In this way, we also discuss the opportunities and challenges that network softwarization may bring to these new paradigms.

The outline of this paper is shown in Fig. 1 and the rest of this paper is organized as follows. We start by introducing some related background knowledge to help build a preliminary view of NFV in Sections 2 and 3, which include the motivation for NFV, basic terminologies, standardization activities and an evolving history of NFV. Section 4 presents an extensive and comprehensive overview of the NFV reference architecture defined by ETSI, and introduces several typical NFV examples as well as the solutions. Section 5 summarizes the algorithms for many popular issues related to VNF. The ongoing research challenges and future directions of NFV are presented in Sections 6 and dummyTXdummy- 7 respectively. Finally, we summarize this work in Section 8.

2. Network state Quo and NFV motivation

In traditional carrier networks, there are a lot of middle-boxes (i.e., proprietary hardware appliances). Generally, middle-boxes indicate the forwarding or processing devices that transmits, transforms, filters, inspects, or controls network traffic for the purpose of network control and management [29]. Hence, they are essential elements to support traditional network services. Typical examples of middle-boxes include NATs that modify packet source and destination addresses, and firewalls that filter unwanted or malicious traffic, etc. To give an overall view, we summarize the commonly used middle-boxes (i.e., routing/forwarding devices, NAT, Wide Area Network (WAN) optimizer, proxy, firewall, Flow Monitor (FM), Intrusion Detection System (IDS), Intrusion Prevention System (IPS), Deep Packet Inspection (DPI), etc.) in Table 2. However, the categories of middle-boxes today are far more than what we mention here and a more detailed taxonomy of middle-boxes can be found in Ref. [30].

Due to the diverse and various requirements on network services, the number of middle-boxes is increasing constantly. Each middle-box appears as a solution for specific purpose. For example, a middle-box of IP firewall is required if a bare computer wants

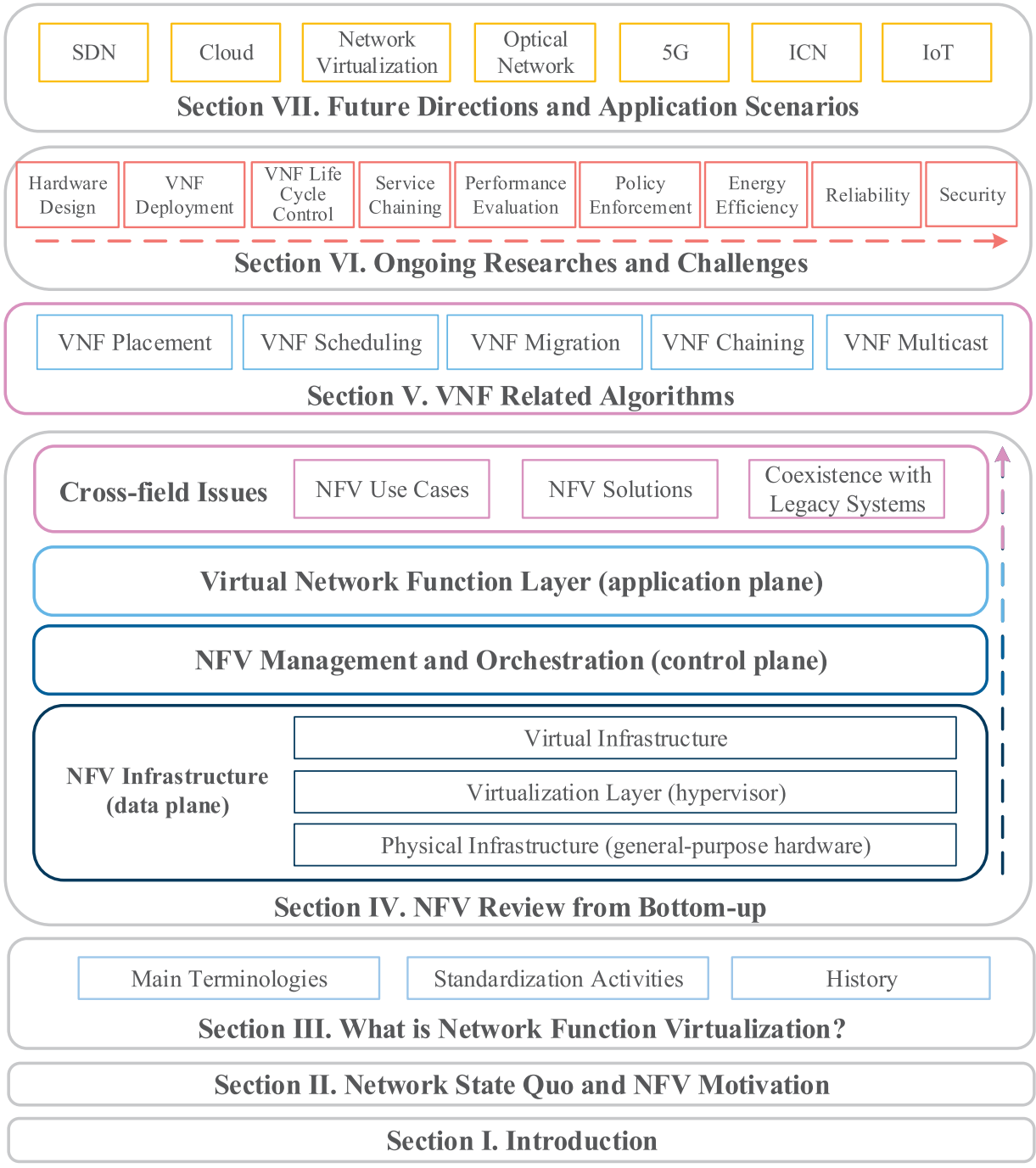

Fig. 1. The condensed structure of this survey.

to be protected from certain threats [35]. However, before applying them into use, these middle-boxes have to be integrated into network infrastructure via a complex deployment process which requires not only a lot of manual work of technically trained person, but also a long deployment cycle. In this way, such purpose-built middle-boxes are certain to cause many issues in the long run [36]. Firstly, due to the fact that middle-boxes are standalone and closed, they naturally introduce new failures when they crash, and the diagnosis for these failures and some mis-configurations would be rather complex [34]. Besides, new middle-boxes may be incompatible with old protocols which are designed without considering the unpredictable appearance of new middle-boxes. In order to make the new middle-boxes work, the reconfiguration or patch procedures are inevitable [33]. As a result, launching a new network service is costly and time-consuming. On one hand, the network operators need to pay for the purchase of new middle-boxes and the maintenance of old middle-boxes, which increase both the CAPEX and OPEX. On the other hand, these new and old middle-boxes do not share the same underlying hardware even with enough available capabilities. Thus, coordinating these middle-boxes is another difficult process that takes long time and high cost on employing technically trained person [37,39].

Apart from the increasing number of middle-boxes, it is known that they are fixed to somewhere in the network and cannot be moved or shared easily. Hence, the network becomes more and more ossified and inflexible [36]. This situation is even worse with

Table 2
Summary of commonly used middle-boxes with corresponding information.

Middle-box	Data input	Actions	Approach
Routing/Forwarding devices [30]	Packet	Routing table update and data forwarding	Protocol enforcement and packet header match.
Load Balancer [31]	Packet, flow	Address rewrite or traffic reroute	Payload match and header match.
Network Address Translator [32]	Packet, flow	IP address assignment or translation	Payload match and header match.
WAN Optimizer [33]	Flow, session	Shaping, dropping, priority alteration	Similarity detect, compression and caching.
Proxy [34]	Flow, session	Mapping, rewrite and routing	Similarity detect, caching, proxy rule expression matching.
Firewall [35]	Packet, flow, session	Allow, bypass, deny and log only	Rule matching, packet filtering, traffic isolation, interception.
Flow Monitor [36]	Packet, flow, session	Monitoring, capturing, logging	Counter, statistics and analysis.
Intrusion Detection System [37] Intrusion Prevention System [38]	Packet, flow	Monitoring, logging, reporting or blocking	Alarm filtering, signature based, statistical anomaly based and stateful protocol analysis detection methods.
Deep Packet Inspection [39]	Packet, flow	Packet classification and content inspection	Port mirroring, rule enforcement, traffic reassemble and content decode.

the increasing number of network services. To solve or at least relieve such situation, some widespread consolidation ideas are first adopted, which include *CoMb* [40], *APLOMB* [41], *INP* [42], *DOA* [43] and *Stratos* [44]. In particular, *CoMb* [40] presented a new architecture for middle-box deployment, which systematically explored opportunities for resource consolidation and addressed the corresponding resource management issue. *INP* [42] implemented a resource consolidation prototype in HP labs, which orchestrated various resources and network devices in a seamless and efficient way for service deployment and provision. Similarly, *Stratos* [44] was also a service orchestrator in charge of orchestrating and managing virtual middle-boxes. Instead of deploying middle-boxes in a local network, *APLOMB* [41] allowed steering traffic to a cloud service provider who offered the required middle-boxes, thus reducing network cost and management burden. *DOA* [43] was proposed as an extension to the Internet architecture. It not only allowed but also accelerated the deployment of middle-boxes by introducing a set of references in packets as persistent host identifiers in order to solve the references delegation issue.

Although these consolidation ideas addressed the problem caused by middle-box to a certain extent, they did not change the nature of the existence of middle-boxes. Therefore, other solutions were proposed, for example, Click [45] aimed at turning middle-boxes into virtualized entities, while Single Root I/O Virtualization (SR-IOV) [46] and Netmap [47] intended to accelerate the I/O by using virtualization technologies. Others, such as the Global Environment for Networking Innovation (GENI) [48] and OpenStack [49], even offered the infrastructure composed by COTS based hardware. Although these efforts did not form a standard, they fundamentally contributed to the appearance of NFV as shown in Fig. 2 [50].

NFV targets on reducing the time to market, decreasing the equipment cost and forming a strong, scalable and elastic ecosystem. Firstly, the time spent on new service evaluation and testing can be shortened by automating the NFVI management and orchestration. In this way, the time to market is also reduced. Secondly, NFV allows network operators and service providers to run software based network functions on COTS hardware rather than purpose-built hardware, thus reducing CAPEX and OPEX. Lastly, NFV ecosystem is expected to be built by transforming the proprietary and expensive infrastructure to general-purpose and cheap infrastructure, on which the software based functions (i.e., VNFs) can be executed. The services used to be constructed by various middle-boxes, can be provided by orchestrating different VNFs, that is, placing VNFs on the optimal Network Point of Presences (N-PoPs) of the network and chaining them in a particular sequence. However, to achieve better network performance, VNFs have to provide at least as good service provision experience as the original proprietary hardware based devices would have provided [50].

The coexistence of NFV and the legacy system is inevitable before the thorough transition to NFV, since we cannot virtualize everything economically and immediately [51]. Compared with the legacy system, NFV introduces a lot of "good" differences [7]. The most obvious one is the decoupling of software and hardware, such that users can choose software and hardware separately according to their preferences. This decoupling also accelerates the evolvement of software and hardware. Besides, VNFs can be deployed dynamically to maximize the resource utilization. Given two kinds of VNFs, they can be executed on the same infrastructure without affecting each other. The practical software instantiation and deployment can be automated as long as the related COTS hardware is ready at specific N-PoPs. Lastly, network operators can adjust the resource allocation according to the actual demands of traffic. In this way, the VNF performance can be managed and controlled in a fine grained and high flexible manner.

The appearance of NFV is also motivated by other situations. The traditional IP multicast can be regarded as a distributed routing problem focusing on constructing the multicast topology, and it is hard to provide a reliable IP multicast due to the fact that the underlying infrastructure is heterogeneous and the link failure discovery is very slow. In the meantime, it is also difficult to prevent members from joining the traditional IP multicast, which leads to security threats. Compared with the traditional IP multicast, the NFV based one can relieve these issues. On one hand, NFV can shield the network heterogeneity by decoupling network functions from the underlying hardware. On the other hand, NFV offers a centralized orchestrator which discovers the link failure and threat quickly, thus enabling a fast recovery. Nevertheless, more and more network functions are virtualized as VNFs. Thus, the VNF Forwarding Graph (VNF FG) becomes an important concept for the purpose of orchestrating these VNFs in a better way and promoting the realization of NFV.

In addition, with respect to the traditional network function migration, it usually requires to migrate the whole object, which is time-consuming and energy-wasting. By decoupling network functions from hardware, NFV enables a more efficient way for migration. Specifically, the traditional network function migration problem is equal to the VNF migration problem in the context of NFV. Considering the features of a VNF, it is possible for us to migrate the stateful information, while the other parts of this VNF can be remotely instantiated on the node after migration. Therefore, compared with the traditional way on migrating a whole object, NFV introduces a more efficient way which can save migration time and

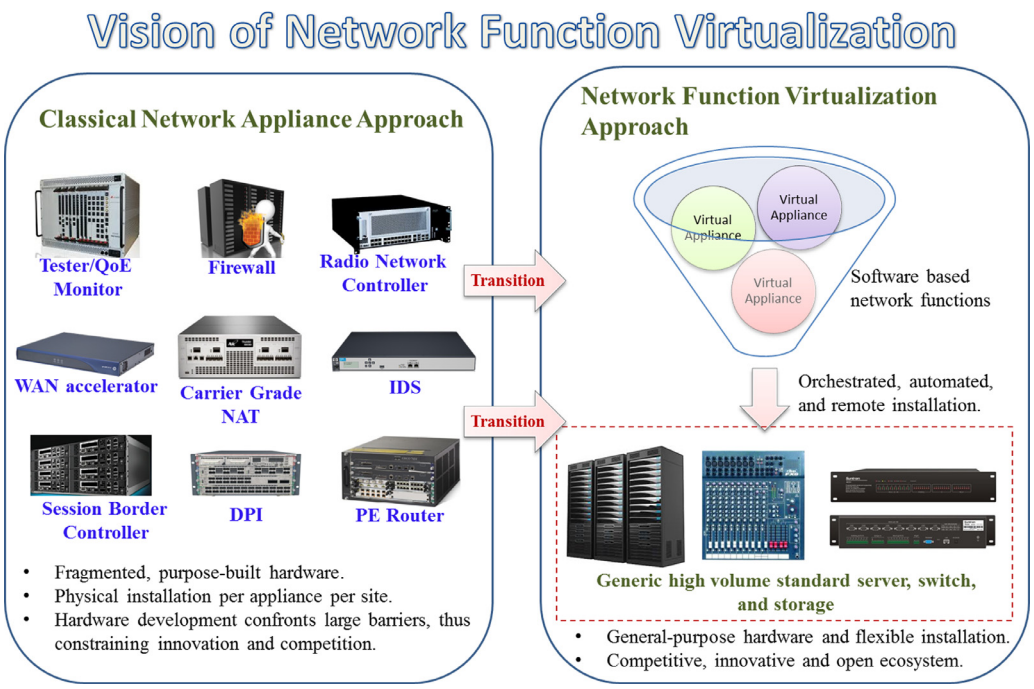

Fig. 2. The vision of classical network appliance to virtual network function [5].

energy greatly [51]. Despite the fact that there are still many other motivations for NFV, we should be aware that it is extremely hard to thoroughly finish the transition from traditional network to NFV network in an economical and immediate way. Hence, a long time of coexistence between legacy system and NFV is inevitable.

3. What is Network Function Virtualization?

The term NFV was originally proposed by over twenty of the world's largest TOs such as AT&T, BT and DT. According to ETSI, the idea of NFV is recognized as a network architecture which transforms the way of building and operating networks by leveraging standard IT virtualization technologies and consolidating proprietary hardware based network functions into standard commercial devices (e.g., x86 architecture machines) [7]. In this section, we intend to provide an intact definition of NFV in order to build a brief impression for readers who are unfamiliar with NFV.

3.1. Main terminologies

NFV has introduced many new terminologies. In order to clearly distinguish them, the imperative ones used throughout this paper are introduced as follows:

1. *Physical Network Function (PNF)*: A PNF is the implementation of a specialized function block with well-defined external behaviors and interfaces. Today, a PNF refers to a network node or a physical appliance, since it is closely coupled with the appliance.
2. *Network Function Virtualization Infrastructure (NFVI)*: NFVI provides a network environment composed of both hardware and software components, in which VNFs can be deployed, managed and executed. One NFVI may traverse multiple geographic places, while the connections among these different geographic places are also considered as part of this NFVI.
3. *Element Management System (EMS)*: An EMS is a set of individual Element Managers (EMs) which are responsible for managing VNF instances in terms of instantiation, execution and deployment during their life cycles.
4. *Management and Orchestration (MANO)*: NFV introduces some new capabilities to the communication network, while MANO is the element used to manage and accommodate these new capabilities. In particular, MANO can be further divided into three entities, that is, Virtualized Infrastructure Manager (VIM), VNF Manager (VNFM) and NFV Orchestrator (NFVO), which are responsible for NFVI management, resource allocation, function virtualization, etc.
5. *Virtual Network Function (VNF)*: VNF is the software implementation of PNF, which has to provide the same functional behaviors and external operation interfaces as PNF does. One VNF may be composed of one or more components. On one hand, if a VNF is deployed in one single Virtual Machine (VM), it is composed of only one component. On the other hand, if a VNF is deployed across multiple VMs, it is composed of multiple components, where each VM hosts one component. Taking the EMS as an example, it is actually a VNF consisting of many individual components (i.e., EMs) which are distributed in different VMs.
6. *Network Point of Presence (N-PoP)*: N-PoP indicates the location where PNF and VNF are implemented. One can access the corresponding resources such as memory and storage from N-PoPs.

In order to help understand the relationships among these terminologies, we connect them in Fig. 3 in a layered structure and highlight them with blue color. Apparently, the co-existence between VNF and PNF is inevitable. The EMs are in charge of VNFs while NFVI is responsible for managing the resources and N-PoPs allocated to VNFs and PNFs. The information (e.g., location) of both PNFs and VNFs are gathered by MANO for the purpose of providing a global view which can be used to coordinate PNFs and VNFs for fast and cost-effective service provision.

3.2. Standardization activities

In order to accelerate the deployment of NFV, many standard NFV activities have been carried out by Standard Development Organizations (SDOs) such as ETSI [5] and OPNFV [52]. Most achievements obtained by these SDOs are open source, such that peer

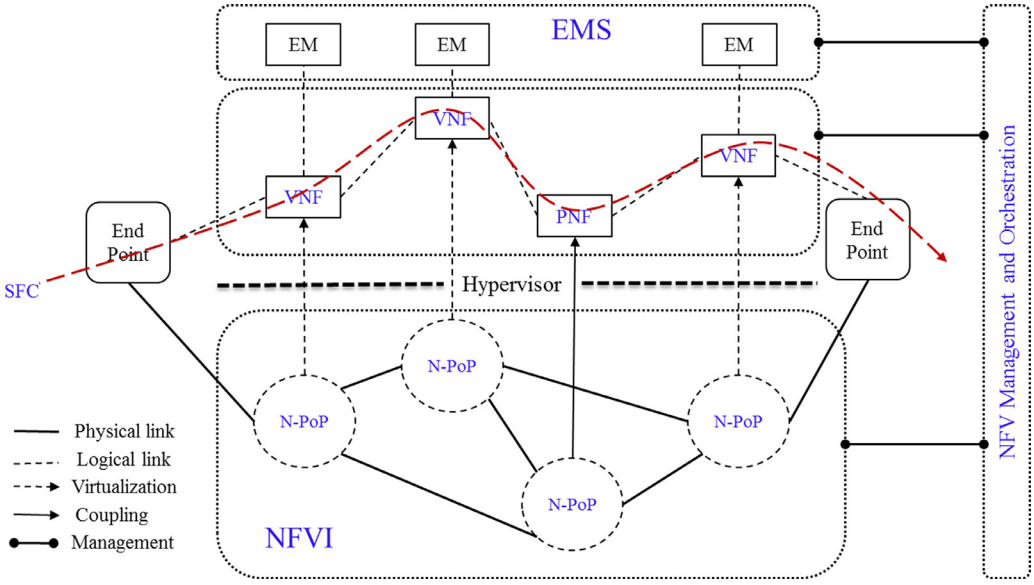

Fig. 3. The relationship of different terminologies within an end-to-end service [53].

workers can learn from their experiences and lessons. This consequently facilitates the development of NFV. Therefore, we introduce the activities of key SDOs that have been contributed to NFV in the following respectively.

3.2.1. European telecommunication standards institute

ETSI has firstly established a research group for NFV in October 2012, that is, NFV ISG [5]. In the same year, the first version of NFV white paper was released by NFV ISG. With more and more members joining in this group, the NFV white paper was updated twice in October 2013 and 2014 respectively. One great achievement of ETSI NFV ISG was the fulfillment of the phase one work with 11 specifications [53–63] released in the beginning of 2015. In addition, the phase two work of NFV was complete with agreement on some objectives like interoperability, while the phase three work of NFV is still under preparation. In the meantime, another group named Open Source Management and Orchestration (OSM) was also established in ETSI in February 2016. Unlike NFV ISG focusing on the overall architecture of NFV, OSM aims at delivering an open source MANO stack based on open source tools and procedures. Considering the relationship between MANO and NFV, OSM actually offers complementary work to NFV ISG toward developing an open source NFV management and orchestration software, and vice versa [64].

3.2.2. Open networking foundation

Although Open Networking Foundation (ONF) intends to accelerate network innovation through SDN, it cannot ignore the significant affection that NFV may have on the future networking. Thus, based on the important value that NFV brings to industry, ONF starts to keep an eye on NFV by publishing a brief solution on how SDN can be used to enable NFV [65]. Then, ONF releases a technical report which illustrates the complementary relationship between SDN and NFV in detail from the perspective of networking architecture [66].

3.2.3. Internet research task force

The Internet Research Task Force (IRTF) has established an NFV Research Group (NFVRG) [67] to carry out researches on NFV. One goal of NFVRG is to provide a common platform, on which the researchers and communities all over the world can share and explore their knowledge about this new research area. Besides, NFVRG also holds workshops or seminars at some well-known conferences (e.g., GLOBECOM).

3.2.4. Internet engineering task force

The Internet Engineering Task Force (IETF) [68] has set up a Service Function Chain Working Group (SFC WG) [68] which focuses on designing SFC architecture in terms of SFC protocol description, Service Function Path (SFP) calculation [69], etc. Generally, the information of SFP is embedded in packet headers [68]. By using such information, the SFC traffic can be easily steered through the required network functions in order before reaching the destination. Besides, the management and security are also within the scope of SFC WG when designing the SFC architecture.

3.2.5. OPNFV

OPNFV [52] is an open source and carrier-grade project. One purpose of OPNFV is to facilitate the development of new NFV services and products. In addition, OPNFV also concentrates on building an open and standard NFV platform for industries via gathering the work from other open source projects (e.g., Open vSwitch (OVS) [70] and Linux Kernel-based Virtual Machine (KVM) [71]), device vendors (e.g., Huawei and Cisco) and standard bodies (e.g., ETSI and IETF).

3.2.6. Alliance for telecommunications industry solutions

Alliance for Telecommunications Industry Solutions NFV Forum (ATIS NFVF) [72] is a North American telecom standard group which focuses on developing NFV specifications that are either complementary or extensive to existing works. Particularly, the scope of ATIS NFVF includes technical requirements, capability catalog and SFC. Besides, ATIS NFVF has an alliance relationship with ETSI on implementing *i*) the common NFVI architecture for supporting flexible VNF deployment, *ii*) the NFV solution for supporting fast service roll-out [24].

3.2.7. Broadband forum

Broadband Forum (BBF) [73] is dedicated to develop broadband network specifications. By introducing NFV into broadband networks, BBF can explore more solutions for service chaining and multi-service implementation in broadband networks [74].

Table 3
The NFV standardization efforts of multiple SDOs.

SDOs	Description	Areas	Work on NFV
ETSI ISG NFV [7]	Industry based NFV specification group.	NFV	NFV terminology, framework, infrastructure and use cases, MANO and VNF, security and trust, resilience and service quality metrics.
ONF [77]	User-driven and open organization.	SDN, openflow	Openflow-enabled SDN architecture to support NFV solution, relationship description between SDN and NFV.
IRTF NFV RG [67]	NFV research group.	NFV, SDN, T2T	Policy based resource management, VNF performance model, service verification, security and resiliency.
IETF SFC WG [68]	Service function chain working group.	NFV, SDN, IPv6	Policy architecture and framework for NFV infrastructures, resource management of service chain, verification of NFV services.
OPNFV [52]	An open source project aiming at facilitating the development of NFV.	NFV	Initial building of NFVI and VIM component, developing VNF applications and use case based testing.
ATIS NFV Forum [72]	A forum for industry NFV solutions.	NFV, 5G, IoT	Defining associated use case, architecture and requirements that emphasize the benefits of NFV in a multi-administrative domain environment.
Broadband Forum [73]	Industry based organization which focuses on broadband networks.	Broadband network, NFV	Collaborating with the ETSI to channel fully interoperable NFV solutions and NFV proof of concepts.
DMTF OVF [75]	A packaging format for software to run in virtual environments.	Cloud, NFV	NFV infrastructure management, alliance with ETSI NFV.
3GPP SA5 [76]	Telecom management working group of 3GPP.	RAN, CN, IMS	In cooperation with ETSI on the management of virtual 3GPP network functions.

3.2.8. Open virtualization format

The Open Virtualization Format (OVF) [75], defined by Distributed Management Task Force (DMTF), not only addresses the virtual appliance portability issue, but also enables managing VNFs in a more automated and secure manner.

3.2.9. The 3rd generation partnership project

The 3rd Generation Partnership Project Service and System Aspects Working Group 5 (3GPP SA5) [76] targets on exploring potential research aspects of NFV. Besides, 3GPP has been working with ETSI on developing normative NFV reference point specifications. One side of the reference point is a 3GPP-defined entity and the other side is a functional block defined by ETSI.

3.2.10. Summary

The above mentioned key SDOs and forums that contribute to the standardization of NFV are summarized in Table 3 and we present a time line of these activities in Fig. 4 to show their sequences.

3.3. History

NFV is still in the infancy. However, the virtualization technology that plays an important role in NFV has evolved for many years. Currently, there are many kinds of virtualization technologies, for example, hardware virtualization in hypervisor of VMware, computing virtualization in cloud. The virtualization of network components (especially network functions) is the primary trait of NFV. In order to give a historical perspective, we start with a small detour of virtualization technology and then make the transition to specific NFV history.

The virtualization terminology was first proposed by Christopher Strachey [78] as a theory in 1959. In the middle of 1960s, IBM proposed its experimental system M44/44X which first introduced the concept of VM [79]. With the time passing by, the capabilities of computer, such as Central Processing Unit (CPU) and Random Access Memory (RAM), were promoted greatly, which could meet the demand of virtualization technologies. Based on this, VMware corporation presented the first commercial virtualization product based on $\times 86$ architecture in 1999 which mainly focused on isolating resources in one server to create independent working environments [80]. Then, the essential requirements of managing all the separated environments resulted in the appearance of various kinds of hypervisors or Virtual Machine Monitors (VMMs), such as Xen [81] of Citrix, Hyper-V [82] of Microsoft and vSphere [83] of VMware.

As explained, the above mentioned virtualization technologies are typically used for resource virtualization, thus to improve resource utilization further. However, taking the practical network conditions into consideration, we can notice that the network is full of proprietary hardware based middle-boxes which introduce not only various kinds of network functions, but also network ossification. To address or at least relieve such case, the virtualization of these proprietary hardware based network functions is extremely required. Under such background, the concept of NFV appeared in the white paper [5] co-authored by over twenty of the world's largest TOs in October 2012. This white paper attracted the attention from both industry and academia. At the same year of November, seven of these TOs decided to choose ETSI as the home of NFV and established the NFV research group ISG. ETSI NFV ISG has grown to 235 companies including 34 service provider organizations and has held seven plenary meetings spanning Asia, Europe and North America by January 2013 [5]. In October 2013, ETSI has updated the white paper [6] and published the NFV architecture framework which identified NFV system components and interfaces among them. Within the next two years, a lot of experts and companies (i.e., 289 companies exactly according to ETSI report in 2015) continually joined the working group (i.e., ETSI NFV ISG) to help promote the development of NFV standards and to share their thoughts about NFV. Currently, the work progress of NFV has stepped into the third stage.

With respect to the first stage work of NFV, it finished successfully at the end of 2014 with the above mentioned 11 specifications. These publications are on the basis of the first released documents of NFV in October 2013 and cover many aspects of NFV. Specifically, they are the management and orchestration [55], architectural framework [56], infrastructure overview [57] (including descriptions of compute domain [58], hypervisor domain [59] and network domain [60]), service quality metrics [61], resiliency [62], and security and trust [63]. All these efforts are achieved after two years' intense work by more than 240 organizations.

With respect to the second stage work of NFV, it is an enhancement and improvement of the first stage work of NFV. Since the architectural work is not considered in this stage, the NFV architecture framework and the MANO framework remain almost the same. In addition, the ETSI NFV ISG has fulfilled most of its initial

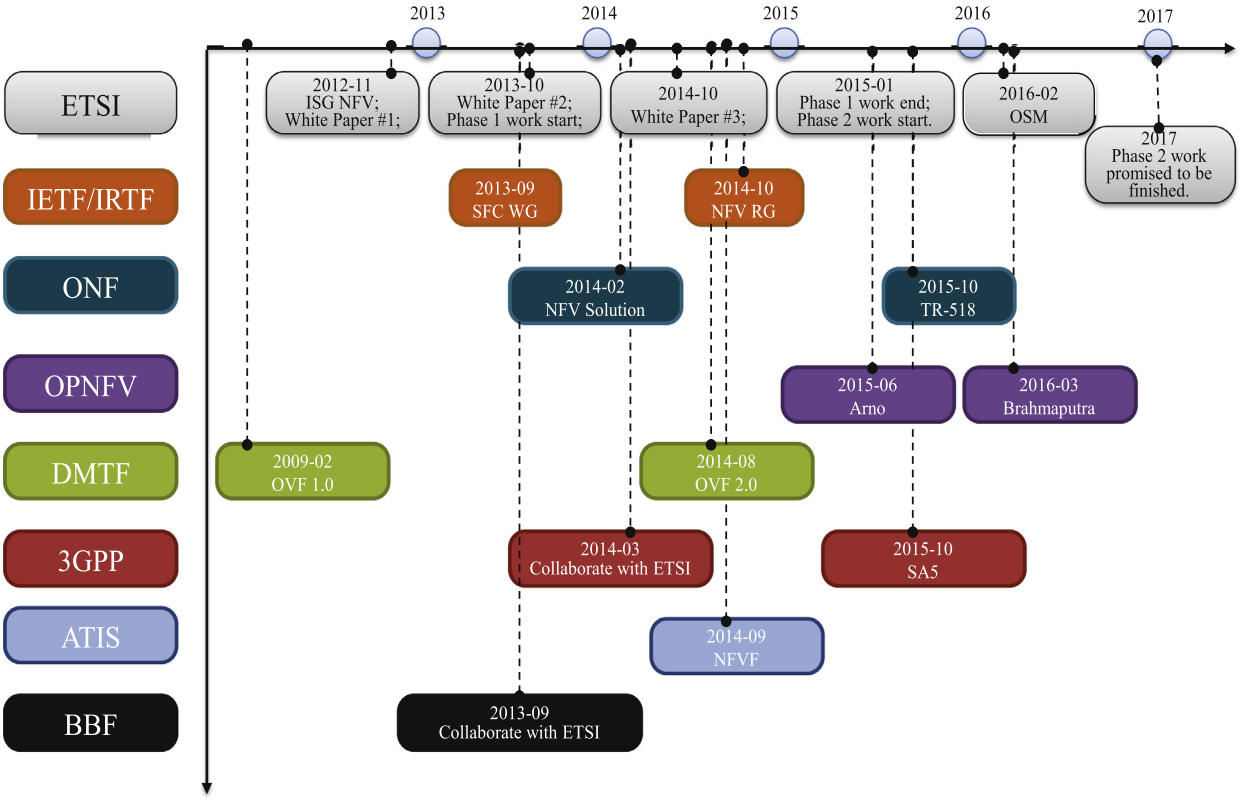

Fig. 4. Standardization SDO activities and main events of NFV.

commitment for the second stage (in 2015 and 2016) to produce the initial set of normative documents, which include the requirements for interfaces identified in the NFV architecture framework, the interface specifications and their information models [84]. In order to validate the second stage work in terms of the management on service descriptors and software images, another work is announced, that is, NFV Plugtests (the first version) which aims at verifying early interoperability between different implementations and main components inside the NFV architecture [85].

Currently, the third stage is progressing on track for delivering a big set of specifications throughout the following years, among which, three new features should be noticed, that is, *i*) the definition of cloud-native VNFs which can fully exploit advantages of cloud computing; *ii*) the support for Platform as a Service (PaaS) model which can be used to assist VNFs following cloud-native design principles; and *iii*) the support for NFV MANO services across multiple administrative domains [86]. The main events of NFV can also be found in Fig. 4.

4. NFV review from bottom up

The NFV architecture is depicted in Fig. 5. In particular, NFVI corresponds to the data plane, which forwards data and provides resources for running network services. MANO corresponds to the control plane, which is responsible for building the connections among various VNFs and orchestrating resources in NFVI. The VNF layer corresponds to the application plane, which hosts various kinds of VNFs that can be regarded as applications. We introduce the three parts from bottom up, with their core responsibilities and technologies discussed respectively. Additionally, in order to present a comprehensive review of NFV, the NFV use cases, solutions and coexistence with legacy systems are also discussed.

4.1. Network Function Virtualization infrastructure layer

NFVI provides fundamental services for fulfilling the objectives of NFV [57]. By deploying a set of general-purpose network devices in distributed locations, NFVI can satisfy various service requirements such as latency and locality, and reduce the network cost on CAPEX and OPEX. Based on the general-purpose hardware, NFVI also provides a virtualization environment for VNF deployment and execution. Although the architectures of current NFVIs are generally the same, their actual implementations may differ a lot. According to the bottom part of Fig. 5, the reference architecture of NFVI can be further divided into three distinct layers, that is, physical infrastructure, virtualization layer and virtual infrastructure. Each of them is introduced in the following subsections respectively.

4.1.1. Physical infrastructure layer

The physical infrastructure of NFVI is composed of general-purpose servers which supply the basic compute and storage capacities. In particular, the servers providing compute capacity are called compute nodes, while those providing storage capacity are called storage nodes. Besides, any compute node can communicate with other network elements through internal interfaces. Therefore, the underlying devices can be further divided into three kinds, that is, compute, storage and network hardware which are introduced respectively as follows.

1. Compute hardware
   The compute hardware refers to the general-purpose compute nodes which are managed by the internal instruction set. Each compute node can be realized in the form of a single-core or multi-core processor (i.e., CPU) in the context of NFV [57]. Currently, there is a rich diversity of servers which can be used as the general-purpose compute nodes. According to their char-

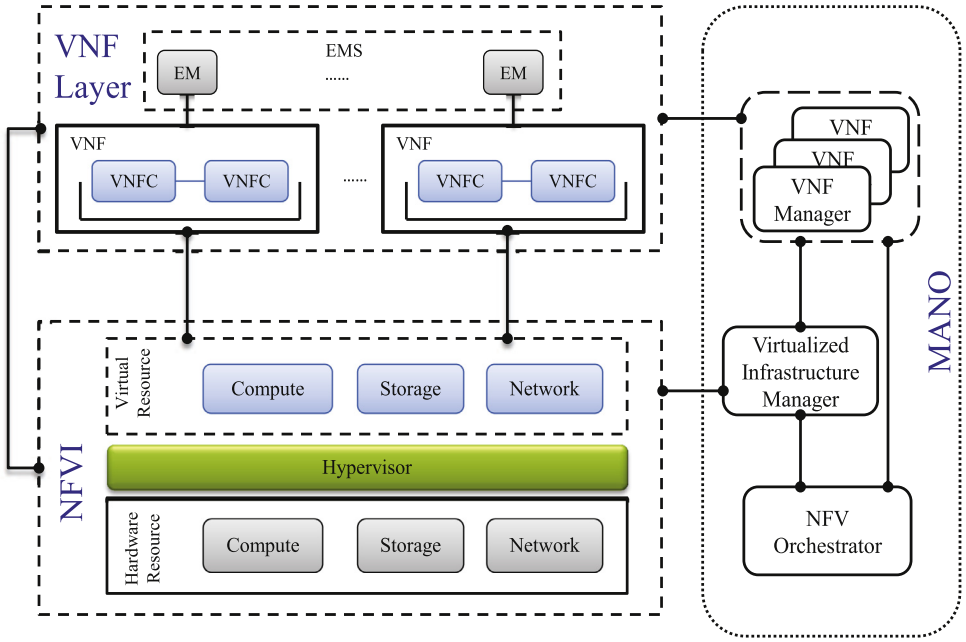

Fig. 5. ETSI NFV reference architecture [56].

acteristics, they are generally classified into the following categories.

- Tower server: it refers to a standalone computer that is built in an upright cabinet known as the tower. Generally, tower servers are constructed with a certain extent of robustness considered in order to reduce service downtime and prevent possible damages. For the deployment of NFVI, most operators would like to use branded servers from major manufactures like Cisco, IBM, Huawei, HPE, etc. However, due to the large volume and weight of tower server, the floor space may be a big limitation for NFVI expansion. In addition, since tower servers are actually independent from each other, each of them will need a complete auxiliary system which includes an individual cooling system, monitor, I/O devices (e.g., keyboard), etc. Such situation naturally leads to high CAPEX for NFVI deployment and maintenance.
- Rack server: it refers to the server that is mounted inside a rack (a shelf to manage servers). Compared with the tower server that one tower only contains one server, a rack can contain multiple servers stacking one above the other, which not only reduces the required floor space, but also consolidates network resources. Currently, the commonly used industry standard servers are 1U or 2U (U is the rack unit and 1U indicates 19 inches wide and 1.75 in. high [87]) rack servers. Due to the fact that many efforts from the open source projects (e.g., CloudNFV) are contributed to the combination of NFV and cloud, most manufactures are transforming their rack systems to cloud based infrastructure, such that they can support and serve NFV workloads in a better way.
- Blade server: it evolves from the concept of rack sever. The blade servers are typically placed inside a blade enclosure to form a blade system that meets the IEEE standards of rack units [87]. Compared with tower and rack servers, the blade server allows more processing power in less rack space, since it shares certain elements of hardware among blade servers within the same enclosure. Considering the case that the NFV workloads are primarily about computing and networking (e.g., packet processing and communication) rather than storage, the blade server is a good choice, because multiple blade servers can be packed into one chassis to provide a high-density server which takes care of networking, power, cooling and hardware management for the entire set of compute nodes.
- Hyper-converged solution: It consolidates the compute, storage and network resources in one box, thus achieving high scalability by dynamically adding or removing such boxes. Although this mechanism brings many benefits such as availability, security and backup, it reduces the flexibility of network deployment, configuration, scaling and upgrading due to the highly convergence characteristics, which may in turn cause hardware utilization and performance issues [88]. Despite this, there are still many vendors working on introducing the hyper-converged solutions into the NFV workload especially for the I/O centric network service provision.

2. Storage hardware

The storage hardware refers to the device capable of storing information temporally or permanently. Many network functions operate based on the storage hardware, for example, the network cache for video streams. Besides, considering the extremely large amount of data in the network, large volume and high speed storage devices are required for data storing and processing. Typically, a storage server is a 2U or 4U box with a large number (e.g., 60+) of Solid State Disks (SSDs) or Hard Disk Drivers (HDDs), and a small amount of assistant compute power and memory. In addition, these storage servers can be expanded by connecting to external disks or drivers. In particular, these storage devices are usually used in the following three aspects.

- Direct Attached Storage (DAS) [89]: it indicates the storage attached to servers via a direct communication path and such storage can only be accessed by the directly attached server.
- Network Attached Storage (NAS) [90]: it indicates a storage device that provides a file access to heterogeneous computers across the network, that is, the file is shared among these computers.

- Storage Area Network (SAN) [91]: similar with NAS, SAN also provides access to shared data storage. The difference is that SAN shares data in block unit as opposed to the file unit of NAS.

3. Networking hardware
   The networking hardware usually comes in the form of proprietary L2/L3 switches or bare-metal switches. In the context of NFV, these proprietary devices are gradually replaced by the industry standard ones which *i*) only support conventional routing protocols; *ii*) only support OpenFlow protocol; *iii*) support both conventional protocols and OpenFlow protocol. Currently, there are many industry standard switches on the market, e.g., IBM RackSwitch [92] and Juniper QFX series switches [93]. The Network Interface Card (NIC) is another important component of networking hardware. Specifically, it is a circuit board or card installed in a switch or a compute node to provide physical connection with other network elements, such that the VNFs in different locations can communicate with each other.

NFVI can be regarded as the data plane of NFV. The NFVI performance can be improved by using either the CPU enhancement [94] or the hardware offload [95] mechanisms. In particular, the former can be fulfilled by plugging the hardware accelerators into the standard COTS servers to accelerate packet processing speed, or supporting huge page to reduce the lookup time. The latter can be fulfilled by using smart NICs to average the load, or adding other co-processors (e.g., FPGA) to accelerate data processing. Nevertheless, due to the trend that the proprietary hardware is going to be replaced by general-purpose hardware, a lot of software based acceleration standards (e.g., P4 [96]) and tools (e.g., SR-IOV [46]) have emerged, which will be introduced in the virtual infrastructure layer.

4.1.2. Virtualization layer

The virtualization layer of NFVI locates between physical infrastructure and virtual infrastructure. Thus, it is regarded as a software platform which leverages the hypervisor to split up physical resources and constitute isolated environment (e.g., VM). Although all these isolated environments share the same underlying infrastructure, each of them is equipped with all necessary peripherals (e.g., NIC) independently [97]. Besides, a lot of network behaviors may happen in the virtual network environment, such as VM instantiation, removal, online migration and dynamic scaling. In order to support these behaviors, the hypervisor can dynamically adjust the mapping relationship between physical resources and virtual resources allocated to VMs, such that a high-level portability among VMs can be achieved. Essentially, the hypervisor can emulate almost every piece of the hardware platform [98]. For instance, the emulation of CPU instruction set would make VMs believe that they are running on different CPU architectures while in fact they are sharing the same hardware platform. However, the practical CPU cycles would inevitably be enlarged when emulating virtual CPU cycle, and this naturally results in the loss of performance.

Typically, the hypervisors used in SDN can also be applied to NFV. For example, FlowVisor [99] is one of the early technologies to virtualize the SDN network. The basic idea of FlowVisor is to share the same infrastructure among multiple logic networks. The NFVI is composed of COTS hardware, which makes the virtualization of hardware resources easy for FlowVisor. There are also many other hypervisors for SDN such as OpenVirteX [100], AutoSlice [101] and AutoVFlow [102]. These hypervisors can be regarded as proxies between data plane and control plane, which provide an easy way for controllers to manage and control the underlying forwarding devices. Nevertheless, we should be aware that the hypervisor in SDN resides between data plane and control plane, while in NFV, it resides between physical infrastructure and virtual infrastructure. According to such difference, it may be concluded that the attention of NFV hypervisor should be focused on the physical resource and Network Function Virtualization. Currently, the mainstream hypervisors used in NFV include Linux KVM [71], Citrix Xen [81], Microsoft Hyper-V [82] and VMware ESXi [83] which are explained as follows.

- KVM [71]: KVM is an open source hypervisor used widely in Linux based Operating System (OS) (e.g., Ubuntu) which is usually called host OS. Correspondingly, those running in the VM are called guest OSs. Based on this, KVM is categorized as the type 2 hypervisor, since it runs on top of an OS. Besides, in order to be ready for deadline-oriented applications and time-sensitive workloads, an extension of KVM is developed, that is, real-time KVM which allows the VM to host a truly real-time OS. On one hand, KVM is a virtualization technology which provides implementation solutions for virtualized network functions decoupled by NFV from the proprietary hardware. On the other hand, the real-time KVM can be used to implement the virtualized network functions with more rigorous requirements (e.g., low latency demanded video streaming).
- Xen [81]: Xen is also an open source hypervisor that allows to execute multiple virtual guest OSs simultaneously on a single physical machine. Unlike KVM that runs within a host OS, Xen runs directly on top of the physical machine. Thus it is classified as the type 1 hypervisor (or bare-metal hypervisor). In this way, Xen enables to run many instances of the same OS or different OSs in parallel on a single machine. These OSs are aware that they are virtualized and do not need the virtualized devices, but have to make special calls to the hypervisor for accessing physical resources (e.g., compute, storage and network). Moreover, Xen supports both the para-virtualization (a lightweight virtualization technique that requires para-virtualization enabled kernels and drivers) and full-virtualization (using virtualization extensions from the host CPU to virtualize guest OSs).
- Hyper-V [82]: Hyper-V is a commercial hypervisor which is integrated in the windows servers of Microsoft and offers a carrier-grade virtualization for enterprises with data centers or clouds. Therefore, for those who want to virtualize the data center workload or build the cloud, Hyper-V may be a common option. In addition, Hyper-V can be installed as a standalone server to provide a set of integrated management tools which can be used in NFV. For example, a virtual network manager based on Hyper-V can be used to create, configure and delete various virtual elements (e.g., VNFs). Besides, a Hyper-V based virtual switch enables NFV with capacities such as tenant isolation and traffic shaping.
- ESXi [83]: ESXi is a type 1 hypervisor that virtualizes enterprise-class servers so that customers can consolidate their applications on less hardware. Although ESXi offers many advanced features for management and administration such as high availability and fault tolerance, it is much more expensive than other hypervisors like KVM. In order to process application workloads with high performance, ESXi tries to maximize the I/O throughput by using few CPU cycles and less power to fulfill the requirements of a wide range of application workloads. However, the workload fulfilled by ESXi may be different from those in NFV which are sensitive to latency, jitter, etc. Therefore, ESXi should be carefully tuned in order to satisfy the NFV workload and maximize the performance.

Apart from the hypervisor, the container technology [103] is another feasible solution for NFV virtualization. The differences between container and hypervisor are shown in Fig. 6, in which we can observe that the container does not need separated systems to host applications or VNFs. Thus, the container can save more over-

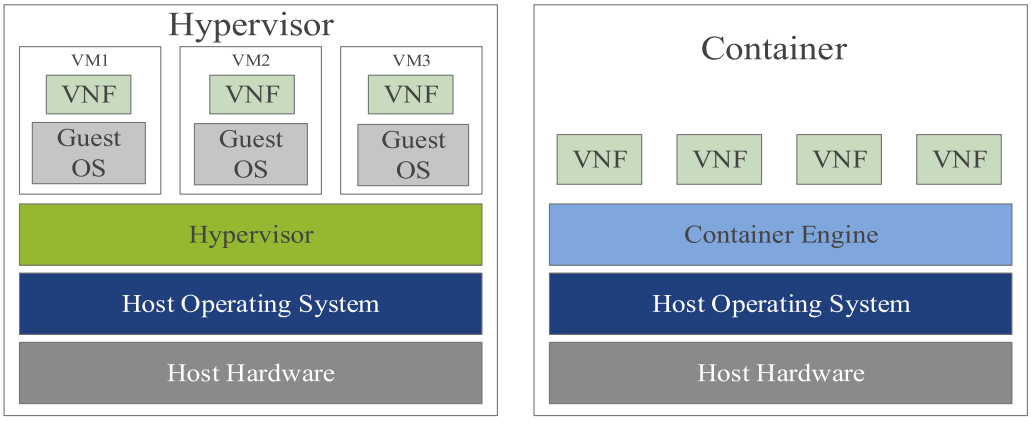

Fig. 6. Comparison between hypervisor and container [103].

head compared to the hypervisor, because the former runs directly on the host OS while the latter runs on the guest OS. In this way, the NFVI virtualization layer is likely to be expanded to include the OSs that based on container engines. Although the container offers a great opportunity to reduce the overhead, it also introduces many potential security issues due to the lack of isolation from the hosts. One representative example of container is Docker [103] which is an open platform for developing, shipping and running applications. It separates the application from infrastructure such that the software based applications can be delivered quickly. In addition, FlowN [104] is a lightweight container based virtualization approach proposed to address the multi-tenancy problem in cloud. Rather than running a separated controller for each tenant, FlowN uses a shared controller platform to run the tenants' applications and each tenant is provided with the vision of its own address space, topology and controller.

The virtual environment provided by a hypervisor or a container must be functionally equivalent to the original hardware environment. Thus, for the same OS, the tools and applications can be used in the virtual environment as long as they are softwarized. Nevertheless, to fulfill the implementation of NFV, many requirements for hypervisors (or containers) still need to be resolved. For example, the hypervisor should offer different granularity in virtualizing network functions (partial or full virtualization), thus to create differentiated services. Then, in order to support the software portability, some well-defined interfaces have to be specified, which include the ones for VNF management and monitoring, and the ones for accommodating PNFs and VNFs due to the coexistence of them, etc. Last but not the least, the existence of hypervisor exposes a lot of potential security threats which need to be carefully considered in the near future.

4.1.3. Virtual infrastructure layer

According to the reference architecture of NFVI shown in the bottom of Fig. 5, the virtual infrastructure layer locates above the virtualization layer and it includes three kinds of virtual resources, that is, virtual compute, storage and networking. These virtual resources play a very important role for providing the virtualization environment in NFV. Nevertheless, we should be aware that they still come from the virtualization of physical resources.

1. Virtual compute
   The virtual compute resource is achieved by the virtualization of hardware processing elements like CPU. Such virtualization operation is usually fulfilled by hypervisors through specific Application Programmable Interfaces (APIs). For example, the libvirt [105] of KVM and the vCenter [106] of ESXi both provide a way for virtual compute resource management. The common part between libvirt and vCenter is that they provide such capability in the form of VM which can be regarded as the execution environment of VNFs. In addition, the Software Defined Compute (SDC) [107] is another kind of compute virtualization technology, which moves the computing functions into cloud and views all the computing resources in one resource pool. Based on this, an on-demand computing resource allocation is obtained through the central interface.
2. Virtual storage
   Similarly, the virtual storage is achieved by the virtualization of storage hardware in the form of DAS, SAN or NAS. In particular, the storage management software is separated from the underlying hardware, which not only enables to create virtual storage resource pools in a flexible and scalable manner, but also brings many additional features such as snapshot and backup. The Software Defined Storage (SDS) [108] is another form of storage virtualization, which creates a virtualized network of storage resources by separating the control and management software from the general-purpose hardware. The Ceph [109] is an example of SDS that provides object, block and file system storage from a single clustered platform. Moreover, the storage network can be used to connect many large storage pools, such that these pools can appear as one single virtual entity.
3. Virtual networking
   The virtual networking is similar to the traditional computer networking. The difference is that virtual networking provides interconnections among VMs, virtual servers and other related components within a virtualized network environment. Although virtual networking follows the physical networking principles, its functions are mostly software-driven, for example, virtual Ethernet adapters and virtual switches. By connecting different VMs, we can build a virtual network for the purpose of production, development, testing, etc. However, we should be aware that these VMs may be on the same hypervisor or across multiple hypervisors. Another important element in virtual networking is virtual switch which allows VMs to communicate with each other using same protocols used in physical switches, without the need for additional networking hardware. In the context of NFV and SDN, the virtual switches are enabled with programmability and flexibility. Thus, a lot of efforts have been devoted to this field. Currently, many open source and commercial virtual switches are developed and we illustrate the representative ones of them as follows:
   - Linux Open vSwitch (OVS) [70]: OVS is a virtual switch that enables a programmable data plane by exposing standard and visible interfaces to virtual networking layer. One important feature of OVS is that it can be ported to other

environments easily, since most of its code is written in C. Besides, OVS can be distributed across multiple physical servers, because it supports multiple virtualization technologies such as KVM and Xen. Compared with the kernel-based switches, OVS can work without a kernel module. In this regard, OVS has a good portability. Besides, with respect to the wide deployment of SFCs, the latest version of OVS is able to support them by integrating the protocol of Network Service Header (NSH) [110] which encapsulates the path identification information in the packet header.

- Linux OpenSwitch [111]: It is an open source and linux-based switching software. Unlike OVS that is deployed over a virtualized or cloud layer, OpenSwitch is deployed in a physical switch. Thus it controls some real hardware ports of the physical switch. Besides, OpenSwitch provides a fully-featured L2/L3 control plane, in which programmability is enabled. OpenSwitch can be regarded as a catalyst for accelerating the move to disaggregated networking switches, such that developers and customers are enabled with specific tools to break the limitations of integrated switches. In this way, the network is controlled in a better way to serve the business needs of customers.
- OpenStack Distributed Virtual Router (DVR) [112]: The DVR is a distributed virtual router in OpenStack. The main feature of DVR is that it moves most of the routing previously performed on the network nodes to the compute nodes. By doing this, it can isolate the failure domain of traditional network node and eliminate the intermediate L3 agents, thus optimizing the performance of network traffic. Moreover, DVR is a supplement of OVS, because it offers routing and gateway functionalities within distributed architecture. Nevertheless, the configurations of DVR only work with the OVS modular L2 driver of Juno [49].
- Brocade Vyatta 5600 vRouter [113]: Vyatta 5600 is one carrier-grade virtual router product proposed by Brocade, which can reach the speed as high as 10+ Gbps. In order to guarantee the performance and availability in NFV, Vyatta 5600 has integrated many features. For example, by supporting nearly all general-purpose virtualized platforms and standard x86 architecture in a resilient and flexible way, the systems based on Vyatta 5600 can easily consolidate resources and maximize appliance utilization with CAPEX and OPEX reduced. In addition, Vyatta 5600 also provides flexible deployment mechanisms for fast and low-cost service provision.

Similar to the hardware acceleration explained in the physical infrastructure layer, there are also many software based I/O accelerators for the data plane of NFV. The representative ones that promote the NFVI performance are illustrated as follows.

- Data Plane Development Kit (DPDK) [115]: DPDK is a set of software libraries and drivers for fast packet processing, which is expected to improve packet processing performance by up to ten times and achieve over 80 million packets per second throughput on a single Intel processor like Xeon. DPDK meets the idea of NFV very well, because it is originally designed to run on any type of processors (e.g., Intel $\times 86$ and IBM Power). In order to reduce the time of new service roll-out from months or weeks to hours, most service providers and equipment vendors are shifting from using different architectures per major workload (e.g., application, control/data plane and signal processing) to implementing all the workload on the commercial standard architecture (e.g., $\times 86$) with DPDK [116]. That is because consolidating all the workload into one scalable and simplified platform makes it easy to offer solutions for the multi-function and multi-vendor problems, for example, the service function chain provision. Based on DPDK, many other software based I/O accelerators such as the extension of DPDK [46], Fast Data I/O (FD.io) [117], IO visor [118] and OpenDataPlane [119] are also proposed, and each of them appears to solve specific limitations of DPDK. For example, the extension of DPDK [46] is used to eliminate the network bottleneck caused by DPDK in the long run, and IO visor [118] is used to extend the kernel functionality and networking stack by residing in the kernel instead of bypassing it.
- Netmap [47]: Netmap is a kind of API implemented in FreeBSD or Linux to accelerate the packet I/O speed. According to Ref. [47], running the packet forwarding applications and libpcap emulation library on top of Netmap can improve the I/O speed to about five times, which can be used to accelerate the performance of NFV and SDN. However, Netmap is based on a very simple data model which may be a double-edged sword. On one hand, this simple data model enables a high portability for applications by using Netmap API. On the other hand, it is hard to accommodate complex and large scale network situations due to the simplicity of the data model.
- Single Root I/O Virtualization (SR-IOV) [46]: SR-IOV is a kind of I/O virtualization technology which aims at maximizing resource utilization and hardware performance. For example, Ref. [46] used the SR-IOV enabled devices to fulfill VNF deployment, which achieved higher packet throughput than using the native Linux kernel network stack. SR-IOV introduces the concept of physical functions and virtual functions which are equivalent to the definition of PNFs and VNFs. The virtual functions are usually lightweight and each of them can be pinned to a VM. Thus, the physical resources are accessed directly and shared by multiple VMs, which helps achieve high performance and avoid resource over-consumption to a certain extent. Nevertheless, the direct access to physical resources would eventually increase the security concerns.
- PF_RING ZC [120]: PF_RING ZC provides a set of APIs to support packet acceleration in a multicore environment. For a system with multiple CPUs, the packet buffer can be allocated in memory regions where a CPU can be accessed directly. Besides, PF_RING ZC clusters procedures such that data can be shared among them. Similar to Netmap, PF_RING ZC characterizes the driver based on a regular Linux driver, which acts transparently unless a special application starts. Once it starts, the regular applications (e.g., packet capturing and monitoring tools) are not allowed to send or receive packets using the respective default system interfaces.
- PFQ [121]: PFQ is a framework built for high speed packet transfer on x86 platforms. However, providing high performance is not the primary target of PFQ. Instead, it intends to offer an easy and safe packet processing, while achieving a relatively high processing speed at the same time. The notable characteristic of PFQ is using a domain specific language to implement the packet processing algorithms. Since it does not rely on specialized drivers, such feature may lead to performance degradation [122].

The virtual network can be implemented in two ways, that is, infrastructure based solution and hierarchy based solution [123]. For the former, it leverages the underlying physical devices to provide the equivalent network functions for the virtual network. This is fulfilled by partitioning the physical network, which does not support the overlay of network addresses. For the latter, it can instantiate several private topologies on the fundamental infrastructure. This is fulfilled by splitting physical network and resources. Comparatively, the latter supports the overlay of network addresses with high management cost and complexity. Considering

the two solutions' features and the practical requirements, enterprises usually adopt the hybrid of them.

4.1.4. Summary

Despite the basic working mechanism of NFVI outlined by ETSI NFV ISG, it does not offer a complete solution for NFVI. Thus, different service providers and equipment vendors begin to build their own NFVI according to their networks' connectivity and geographic distributions. Under such situation, various NFVI solutions and products appear rapidly, because different vendors (e.g., Cisco and Ericsson) have different approaches and standards for the NFVI solution. Nevertheless, these NFVI solutions are mostly assembled from the components introduced in the previous layers (i.e., Sections 4.1.1–4.1.3). For example, the Cisco NFVI solution fully integrates Linux KVM [71] (hypervisor), OpenStack [49] (virtual infrastructure manager) and Ceph [109] (an open-source and software-defined storage system). In addition, no matter how NFVI varies from organization to organization, it must be secure and reliable. Based on this consideration, many critical services are gradually built into NFVI. In order to provide differentiated services, some vendors even try to combine the NFVI components with their existing hardware and software. However, such two trends may result in many different standards, which in turn slow down the deployment of NFV.

4.2. NFV management and orchestration layer

The primary responsibilities of NFV MANO are to manage the entire virtualized context within the framework of NFV. In particular, the context includes virtualization mechanism, hardware resource orchestration, life cycle management of VNF instances, interface management between modules, etc. All the responsibilities are categorized into three parts according to ETSI, namely, Virtualized Infrastructure Manager (VIM), NFV Orchestrator (NFVO) and VNF Manager (VNFM), as shown in the right side of Fig. 5. Specifically, the NFVO is mainly responsible for orchestrating NFVI resources and managing the life cycle of VNFs. In order to provide a network service, multiple VNFs are orchestrated and chained according to the determination of NFVO. However, this turns out to be very complex as we should not only deploy each required VNF (including VNF instantiation and configuration), but also calculate one optimal path to connect them, thus to provide the demanded service. Besides, the VNFM is responsible for managing multiple VNF instances. However, one situation should be noticed, that is, one VNF instance is associated with one single VNFM, while one VNFM may be assigned to manage multiple VNF instances. Most VNFMs are designed to accommodate any type of VNF as well as the management work including VNF instantiation, updating, searching, extension and termination. The VIM manages and controls NFVI resources, such as network, compute and storage. Moreover, VIM can also be customized to handle at least one specific kind of NFVI resource (e.g., network-only, compute-only or storage-only) by exposing the interfaces to the corresponding resource controllers. A WAN infrastructure manager is a typical example of this specialized VIM which builds connectivity among different endpoints within network. Besides, VIM may manage the virtual compute, storage and network resources in NFVI through some of its external interfaces.

It is discovered that the responsibilities of NFVO, VNFM and VIM are partially overlapping. Thus, they are usually implemented as an intact entity. However, ETSI NFV ISG only presents a reference architecture for NFV MANO, which lacks implementation and suffers from many limitations such as security and interoperability. In this way, many works are proposed in order to make a supplementation to the reference architecture and accelerate its deployment. For example, in order to address the security issue of ETSI NFV MANO, the Zero-time Orchestration, Operations and Management (ZOOM) [124], a management and orchestration platform, is proposed to complement the work of ETSI and it offers new security approaches to protect the infrastructure (NFVI), functions (VNF) and services across all layers based on a set of best practices. Besides, ZOOM claims to provide an opportunity for true business agility via zero-touch, self-service and adaptive automation operations.

Unlike ZOOM that focuses on security management and orchestration, OpenMANO [125] targets on implementing the ETSI NFV MANO framework to enhance and accelerate the service provision. In particular, OpenMANO can be further separated into three components, that is, Openmano, Openvim and opemmano-gui respectively. The three components together contribute to a practical implementation of the ETSI NFV MANO architecture. Accordingly, the Openmano is an analogy to the component NFVO while Openvim refers to VIM. Openmano-gui offers two ways for network operation, that is, user-friendly graphical interface and command line interface. In addition, OpenMANO enables using an openflow controller for certain purposes, for instance, constructing the physical network topology in a centralized manner. More importantly, OpenMANO constructs an open ecosystem for telecommunications providers and equipment vendors in order to expand their services and ensure high and predictable performance for the most advanced VNFs. This behavior actually promotes the interoperability which is weak in ETSI NFV MANO reference architecture [55].

Another implementation of ETSI NFV MANO is Open Source MANO (OSM) [126] which is hosted by ETSI. The objective of OSM is to develop open source NFV management and orchestration software stacks. Although OSM has a highly compliant architecture with the reference one, it separates NFVO into two fine-grained components, that is, the service orchestrator and resource orchestrator. Specifically, the former is responsible for end-to-end service orchestration and provision (such service is described by the YAML modeling language and may be composed of both VNFs and PNFs). The latter is responsible for allocating network resources to the determined services over at least one IaaS provider. In fact, the two orchestrators of OSM can be jointly mapped to the NFVO entity in the reference MANO. Besides, within OSM, there is a module for VNF configuration and abstraction, which can be regarded as a generic VNFM with limited features considering its responsibilities. With respect to VIM, the OSM leverages the efforts from OpenVIM, OpenStack and VMware. In particular, OpenVIM is a lightweight VIM implementation, while OpenStack and VMware are alternatives to heavyweight VIM implementations.

Rather than implementing the ETSI defined NFV MANO architecture personally, some works are realized based on existing achievements and efforts. For example, the Tacker is proposed as a subproject of OpenStack. In particular, it offers a solution for VNFM and NFVO, which can be used to deploy network services and operate VNFs on any general-purpose based infrastructure platform [127]. In fact, Tacker relies on the original service orchestration component (i.e., Heat) of OpenStack. From the perspective of architecture, Tacker does not have any difference from the ETSI reference architecture. It includes the components of NFV catalog, VNFM and NFVO. In particular, the NFV catalog of Tacker provides the attribute description of various VNFs and services, the VNFM of Tacker provides basic life cycle management of VNFs, and the NFVO provides VNF placement policies for provisioning services. Beyond that, Tacker also offers some new features, for example, the VNFM enables to auto heal and scale VNFs based on different policies and the NFVO enables the ability to orchestrate VNFs across multiple VIMs and multiple sites (i.e., N-PoPs). Nevertheless, we should be aware that some important aspects are still not covered in Tacker, for example, the missing of PNF orchestration. Considering the inevitable coexistence of VNF and PNF, only orchestrating

VNFs is not enough for improving network performance. Thus, the independent orchestration of PNF and the hybrid orchestration of them are essentially required.

Currently, due to the widespread research on service orchestration, more and more efforts are gradually shifted to the NFVO component, which leads to a situation that the boundary between MANO and NFVO is becoming blurry.

One example that focuses on the orchestration framework is Cloudify [128]. It is originally an open source and pure cloud orchestration framework for managing applications and services, detecting errors and failures, and automatically remediating issues, etc. With the emergence of NFV, Cloudify is expanded to include the primary NFV features. For example, the pluggable architecture of Cloudify enables a fully management and orchestration of VNFs. Specifically, Cloudify is composed of three parts, that is, *i*) the Cloudify Manager which performs deployment and life cycle management of applications; *ii*) the Cloudify Agent which is used to manage workflow execution via an appropriate plug; *iii*) the third-party component (e.g., Logstash [129] and RabbitMQ [130]) which provides overall life cycle management with Cloudify Manager. From this point, Cloudify plays the role of NFV orchestrator within the ETSI NFV MANO reference architecture and cooperates with other NFV components (e.g., VNFM, VIM and NFVI) to better orchestrate the software based networks. In addition, Cloudify introduces the new concept of cloud-native VNFs that can fully exploit the advantages of cloud computing. Considering the tremendous work of cloud computing, the NFV deployment will be accelerated with the assistance of Cloudify.

OPEN Orchestrator (OPEN-O) [131] is a Linux foundation collaborative project and it focuses on establishing an open framework to orchestrate end-to-end services across legacy, SDN and NFV supported networks. Likewise, OPEN-O is compliant with the reference architecture of ETSI NFV MANO. Besides, to guarantee the generality and flexibility of OPEN-O, the standard service modeling language YANG [132] is used by OPEN-O. As explained, OPEN-O provides services across three kinds of networks, and each of them is managed and orchestrated by corresponding orchestrators. For example, the SDN orchestrator is responsible for orchestrating services over SDN and legacy networks, while the NFV orchestrator mainly focuses on orchestrating services composed of VNFs. In order to make such two independent orchestrators cooperate, a global service orchestrator is also defined in OPEN-O. In this way, given a service, it is firstly described in a single global service description. Then, according to specific requirements, it can be decomposed to SDN service description and NFV service description for detailed implementation.

The FROG [133] is one SDN, NFV and cloud integrated orchestration architecture that mainly focuses on supporting heterogeneous infrastructure. Likewise, FROG also divides the infrastructure into different domains and each domain is under the charge of a dedicated FROG domain orchestrator. With respect to the first one, it is a pure SDN supported domain and the SDN controllers (e.g., Floodlight and OpenDaylight) are regarded as the corresponding orchestrator of this domain. The second one is a traditional cloud computing supported domain and the cloud platforms (e.g., OpenStack) are regarded as the corresponding orchestrator. The third one is a resource-limited home gateway domain and it is under the charge of a customized orchestrator. In addition, FROG introduces a global orchestrator to control and coordinate the three infrastructure domains. Then, based on the capacities and resources of each domain, and the constraints specified by services, the global orchestrator can determine how to split the service graph and deploy the separated parts on selected infrastructure domains. Reviewing OPEN-O, although it does not consider the cloud based infrastructure, it adopts the same hierarchical orchestrators with FROG. In this regard, FROG has the same structure with OPEN-O. However, unlike other orchestrators that rely on traditional REST API to interconnect different components, FROG mainly relies on an intermediate message bus (i.e., double-decker) to interconnect the components, which is more efficient and faster than using REST API.

Apart from the above heavyweight orchestrators, there are many lightweight ones. For example, M.T. Beck et al. proposed a lightweight NFV orchestration framework that enabled researchers to simulate their deployment and orchestration algorithms in NFV [134]. Due to its lightweight characteristics, new strategies can be deployed and tested quite straight forward and quickly. However, we should be aware that the scope covered by such orchestrator is also limited, that is, it may not have the ability to run or test the architectural schemes or applications. Another lightweight NFV orchestrator can be found in [135], which focused on providing solutions for managing and orchestrating VNFs quickly and easily. Different from the one proposed in [134], the major selling point of this work was the service scalability. Specifically, it provided a VNF independent approach to integrate any existing VNFs to the provisioned services without making any reconfiguration. In addition, this orchestrator also provided the functionality of VNFM in order to support some de facto standards from SDOs such as OpenStack and OPNFV.

4.3. Virtual network function layer

The VNF layer plays an important role in the whole NFV structure. NFV is intended to abstract the underlying PNFs and finally implement them in the form of software (i.e., VNF). VNFs can provide the network functionalities originally provided by proprietary network devices, and are expected to be executed on the COTS hardware.

The structure of the VNF layer, which may include many isolated VNF instances, is illustrated in the top of Fig. 5. In addition, each VNF is composed of multiple VNF Components (VNFCs) which are managed by the corresponding EMs. All the EMs within the same domain together make up the EMS. The EMS is in fact a kind of VNF rather than part of MANO and it is responsible for managing various VNFs. Likewise, the VNFM is also in charge of managing VNFs in terms of instantiation, update, query, scaling and termination. However, in order to well perform those management functions, the EMS has to collaborate and exchange the VNF related information with VNFMs.

Basically, PNFs inside the network infrastructure, such as border gateway, firewall and dynamic host configuration protocol, must have perfectly defined external interfaces and the corresponding models of behaviors. PNFs provide network functions in physical network, while VNFs play the same role in virtual network environment. In this regard, the work used to be done by PNFs can now be replaced by initializing corresponding VNFs. Besides, chaining multiple VNFs locating in different positions of the network can make up a service chain. Based on the practical requirements of enterprises, the VNF locations can be dynamically selected.

Due to the diversity of VNFs, they may exist in different layers within the NFV architecture. For example, Brocade Vyatta 5600 vRouter [113], a virtual switch, can be regarded as in the virtual infrastructure layer, while OpenDaylight [136], a software based controller, can actually be regarded as in the management and orchestration layer. However, since the two layers are already explained in the previous sections, we mainly focus on illustrating VNFs in the application layer. According to the definition of VNF, all the middle-boxes presented in Table 2 can be virtualized as VNFs. With respect to each kind of VNF, there may be multiple implementations with different features considered. For example, the virtual NAT implemented by VMware provides a way for VMs to communicate with the host while the one implemented by NFWare is extended to the carrier-grade level. Hence, it is hard to illustrate

Table 4
Products and solutions of VNFs.

Vendors	Product/Solution	Description
Infoblox	virtual secure Domain Name System (DNS) [137]	An expansion solution of the original DNS, which reduces the business and operation risk during the network transition to NFV and SDN.
NEC	NFV C-RAN [138]	A cloud based RAN with automate L2/L3 functions adding and removing for NFV to meet the traffic demand in NFV.
NFWare	virtual carrier grade NAT [139]	Centralized network addresses translator.
Oracle	IMS Session Delivery [140]	An agile IMS solution for delivering consumer VoIP and VoLTE services.
Red hat	Linux-Atomic host [141]	A lightweight platform support running applications in Linux containers.
6wind	Virtual Accelerator [142]	Accelerating packet processing for virtual network infrastructure in NFV.
	Turbo IPsec [143]	A software Virtual Private Network (VPN) appliance deployed on COTS servers in bare mental environment with the same functionality of the legacy IPsec VPN gateways.
Cisco	Virtual Port Channel [144]	Allowing physical links between two devices to appear as a single port channel to a third device.
	Virtual Managed Services [145]	A cloud native solution for delivering new software defined WAN services to business customers.
Ericsson	Virtual Router [114]	A carrier-grade software system which offers network operators the ability to deploy services with high agility and performance.
	vEPC [146]	Virtualized Evolved Packet Core networks.
Juniper	vMX series edge router [147]	Revolutionary carrier-grade virtual routing for enterprises and service provider networks.
	vSRX integrated virtual firewall [148]	A virtual firewall designed for enterprises and service providers to achieve capabilities of firewall, security and automation in a virtual machine.
Nominum	N2 [149]	A virtual communication platform depending on browsers.
	Vantio CacheServe [150]	A DNS solution which integrates the N2 platform.
F5	Policy Enforcement Manager [151]	Optimizing and monetizing networks with context-aware policy enforcement.
	Local Traffic Manager [152]	Application delivery with programmable infrastructure in a reliable, secure and optimized way.
	Virtual Edition [153]	Deploying software-defined application service in hybrid, virtual and cloud environments.
Metaswitch	Perimeta Session Border Controller [154]	Carrier-grade virtualized network function which supports large scale communication security.
	Clearwater [155]	IP Multimedia Subsystem in cloud computing.
	Voice over LTE [156]	A VoLTE solution built from the ground up using cloud native service methodologies.
Brocade	virtual Mobile Analytics [157]	Brocade network visibility platform for end-to-end mobile networks.
	SteelApp Traffic Manager [158]	A leading virtual application delivery platform for virtual environments or cloud.
	Vyatta 5600 vRouter [113]	Networking industry leader in virtual router with high performance and scalability.
	virtual Application Delivery Controller [159]	A software based VNF solution for fast, reliable application delivery across the virtual and cloud platforms at massive scale.

them all. Nevertheless, we summarize the representative ones according to their emphasis in Table 4 to show a relatively complete overview of current VNF products.

As explained, the number of VNFs is tremendous as well as their categories. Therefore, it is important to have some automated approaches for building and managing them. One typical example is Click [45] which is designed to build flexible and configurable network functions. Currently, Click can provide many software based functions for packet forwarding and processing. For example, the Click element IPFilter plays the role of firewall, while the Click element IP/UDP/TCP Rewriter can be regarded as a kind of NAT. Apparently, the main characteristics of NFV accord with the idea of Click. In this regard, many researches use the Click modular to simulate a NFV environment. However, considering its limitations, an improvement is proposed, that is ClickOS [160]. Based on the definition of the Click elements, ClickOS presents a high-performance and virtualized software middle-box platform, on which a wide range of middle-boxes are implemented in software. These software middle-boxes include the firewall, carrier-grade NAT, load balancer, etc. Since the document of Click project was not updated any more after 2011, the ClickOS will replace it in the long run. Apart from the two lightweight platforms, there are also many open source and commercial ones for building and managing VNFs, which typically appear as a whole NFV solution that are introduced in Section 4.4.2.

Currently, VNFs can be implemented in two kinds of environments. The first one is the VM environment which can be offered by virtualization technologies such as VMware, KVM, XEN, Hyper-V, etc. The second one is the container environment offered by Docker for example. VM offers an isolated duplicate environment of a real computer machine for running VNFs, while container only includes the necessary elements for running VNFs. Based on VM and container, there evolve many other technologies for VNF implementation, which are compared in Fig. 7 and illustrated as follows:

- VM [161]: According to the layered structure shown in Fig. 7, a guest OS is installed in the VM, which can be used to run VNFs. Besides, as explained previously, the VM runs on hypervisors (e.g., KVM and Xen) which also need a host OS environment running on the physical hardware. Currently, VM is the default environment for running VNFs.
- Container [103]: Compared with VM, the container saves the requirement for a guest OS. Instead, it directly virtualizes the host OS, on which necessary libraries are installed. Such design simplifies the underlying OS and the VNF execution, because containers in one compute node share a kernel. A small container image indicates that the density of per compute node can go up greatly (e.g., by 10 times). Despite the strong desire to run VNFs in containers, there are still many problems required to be solved. For example, the kernel sharing among containers would result in isolation and security issues inevitably.
- Container in VM [162]: This approach runs containers in a VM for the purpose of achieving their benefits at the same time. Specifically, on one hand, the VNF execution is simplified by using the container technology. On the other hand, the VNF isolation can be guaranteed to a certain extent by using VM. However, the respective performance achieved in this approach cannot exceed the case when using one of them alone. Despite this, this approach is a very common way for enterprises that do not require high performance in NFV.
- Clear Container [163]: The clear container is part of the Clear Linux project originated from Intel. The main idea behind clear

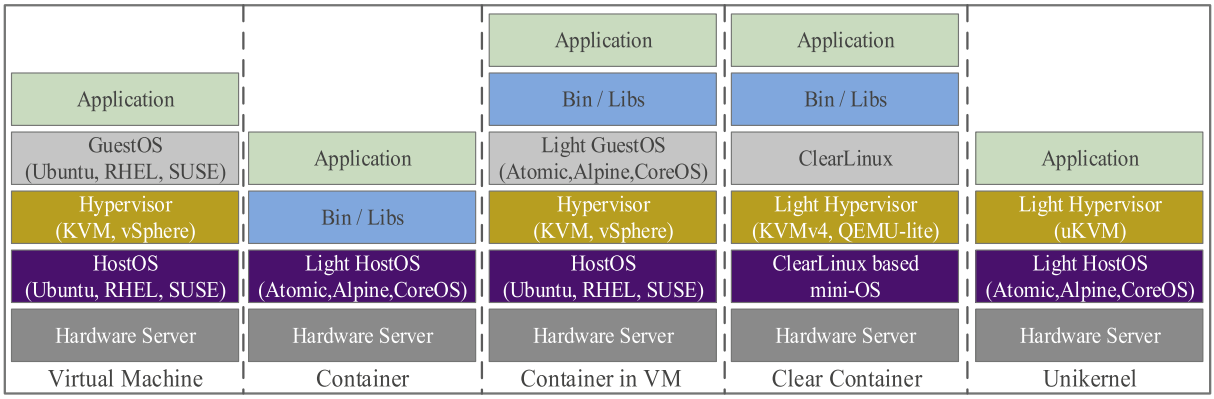

Fig. 7. Virtualization environment alterations for VNFs [165].

container is similar to the case of container in VM, that is, running a container in a VM. However, they differ in implementation, that is, the VM wrapper is shrunk dramatically in this approach in order to make the entire VNF environment appear as a container rather than a VM. Besides, the host OS is replaced by a lightweight one based on Clear Linux. In this way, the corresponding boot-up time and image size are close to those of a container. Considering the popularity of using containers in NFV, this approach can also be leveraged for the development of NFV.

- Unikernel [164]: The unikernels are in fact VMs that constructed by using library operating systems. However, unlike traditional VM that offers an entire guest OS, unikernels only connect VNFs to the libraries they require. Based on this design, the sizes of unikernel images are similar to those of containers, and their boot up times are also similar. Due to the two benefits, the unikernels can be a good option for NFV. However, due to the lightweight characteristics of unikernel, any new VNF has to be recompiled before running in it. Considering the large amount of VNFs and VNF types, this may be a big challenge when applying unikernel to NFV.

4.4. Cross-field issues

Due to the potential of NFV, there evolve many related use cases and solutions for the purpose of accelerating its deployment. Besides, the coexistence between the NFV system and the legacy system is another issue that should be focused on. We illustrate these topics respectively.

4.4.1. NFV use cases

ETSI has proposed nine typical NFV use cases [166] which greatly promote the NFV standards and related products in the first phase of work. These use cases are the main reference examples for most industries when deploying NFV as the networking solution. However, they are difficult to follow. In this section, we redesign four use cases and present them in a relatively simple and easy-to-be-understood way.

- *NFVI as a service (NFVIaaS)*
  Generally, it is almost impossible for any service provider to construct and maintain its own physical infrastructure all over the world, while on the contrary, the user demands are global. Due to such contradiction, the concept of NFVI as a Service (NFVIaaS) appears. In particular, NFVIaaS indicates that NFVI can be offered and sold from one service provider to others. In this regard, it is very convenient and cost-effective for one service provider to serve its customers who reside far away from its own NFVI locations by running the VNF instances remotely on other service providers' NFVI platforms. Although NFVIaaS can facilitate the deployment of NFVI platforms, ETSI NFV ISG only describes a simple use case for NFVIaaS instead of providing a detailed specification. Considering this, IRTF and VMware both elaborate a document about NFVIaaS. In particular, the former depicts an architecture framework for NFVIaaS, which aimed at policy based resource placement and scheduling [167]. The latter focuses on describing a successful transformation to NFVIaaS, during which an operating model and an organization model are proposed [168]. From the perspective of NFVI owners, they have to make sure that only the authorized customers can execute and deploy the VNF instances on their NFVI platforms. Meanwhile, in order to prevent the customers from affecting each other, the thorough customer-isolated and resource-constrained mechanisms are also required. Therefore, a well-designed NFVI is critical for the implementation of NFVIaaS. Currently, most of the researches intend to build the relationship between NFVI and cloud. For example, Ref. [169] demonstrated the software defined infrastructure which was the combination of NFVI and IaaS under the centralized management of SDN. Another important focus is the NFVI benchmark and testing. In Ref. [170], a methodology based on fault injection was proposed for dependability evaluation and benchmarking of NFVI.
  Fig. 8 intends to illustrate a simple example, in which service provider *SP*1 runs its VNF instances on the NFVI platform owned by service provider *SP*2. We simplify the process in three steps as shown in Fig. 8. First, the user *U*1 asks *SP*1 to prepare a certain VNF instance before reaching the destination *U*2 (shown by ①). However, *SP*1 discovers that the user's destination is close to *SP*2, and compulsively deploying the required VNF instance on the NFVI of *SP*1 would cost a great deal. Hence, the better way is to rent *SP*2's NFVI and deploy the required VNF instance on it (shown by ②). After finishing the deployment, the traffic between *U*1 and *U*2 traverses the VNF instance on *SP*2 (shown by ③). Importantly, the NFVIaaS is reflected by step ②. Furthermore, apart from the situation between different service providers, NFVIaaS can also happen between different apartments inside one service provider.
- *VNF forwarding graph (VNF FG)*
  Generally, there are many data centers distributed across large geographies and they may not belong to the same operator. For any data center, it deploys a lot of service nodes at various points in the network topology, on which a variety of layer 4 through layer 7 service functions (i.e., VNFs) are deployed in both physical and virtual forms [171]. In this way, the VNFs hosted at a given service node may overlap with those hosted at other service nodes, and this situation also happens between any two data centers. For example, four different data centers owned by different service providers are shown in Fig. 9, where

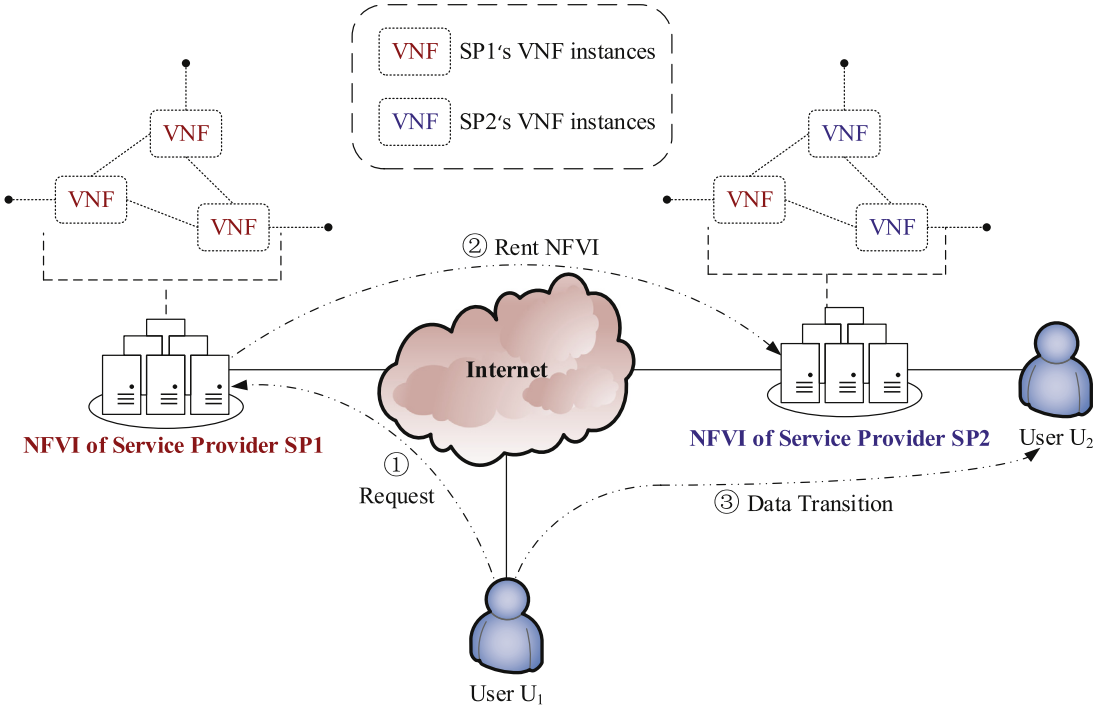

Fig. 8. The use case of NFVI as a service.

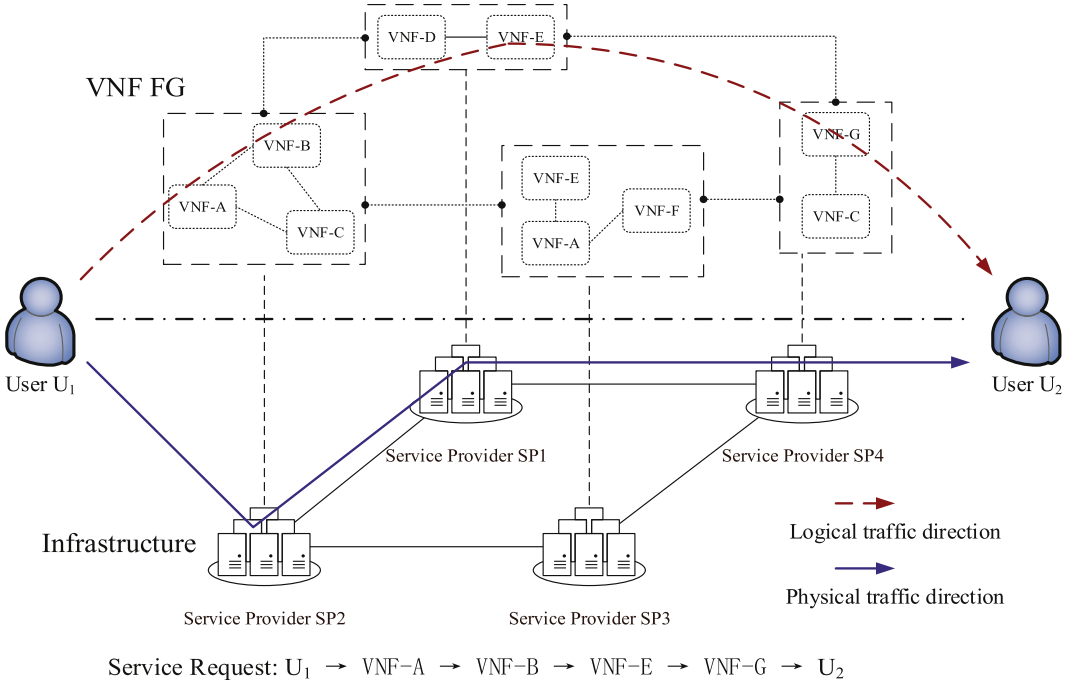

Fig. 9. The use case of VNF forwarding graph.

the second data center and the third data center offer the same VNF (i.e., VNF-A).

Traditionally, the network traffic must follow a defined path to deliver specific services, along which the proprietary network functions are already deployed. A very simple example of network service may happen on one bidirectional point to point link. However, we must be aware that the service delivery in virtualized network environment is much more complex. Likewise, in Fig. 9, the network functions supported by the four data centers are abstracted in the form of VNFs, with the interconnection among them remained. In particular, all these abstracted VNFs constitute to the graph which is referred to as VNF Forwarding Graph (VNF FG) [172]. Thus, VNF FG can be regarded as an analogue of physical network forwarding graph that connects physical appliances via bidirectional cables, which actually connects VNFs via virtual links for the purpose of describing the traffic among these VNFs. In particular, connecting multiple VNFs in sequence can constitute a service which is referred to as SFC in the context of NFV. However, to deploy and implement this SFC, we first need to determine the

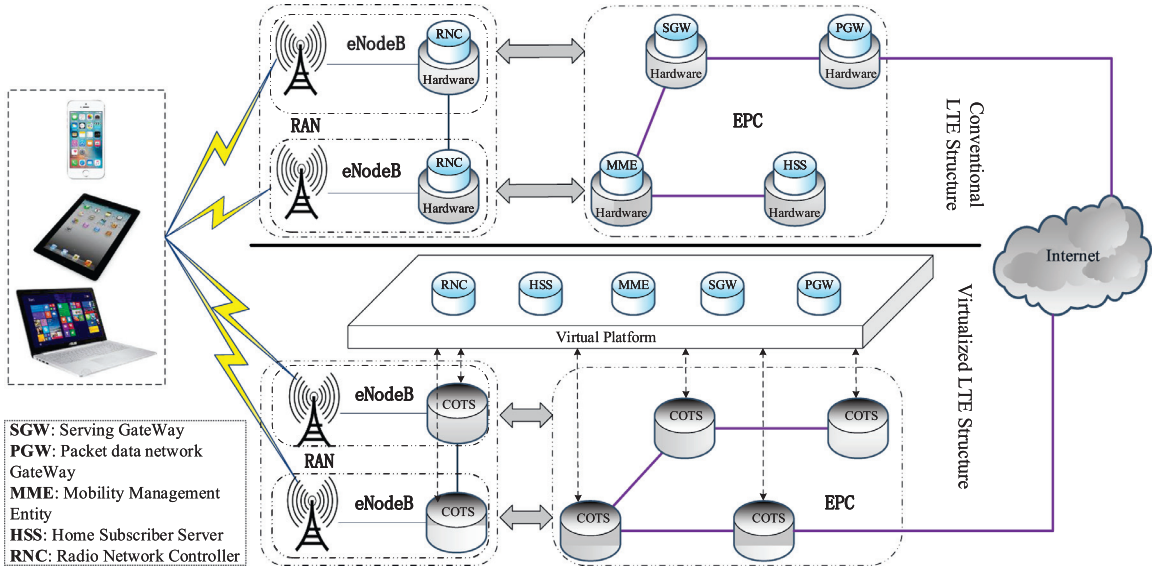

Fig. 10. The use case of long term evolution virtualization.

routing logic of traffic among these connected VNFs. This process can be fulfilled by VNF chaining algorithms which are detailedly discussed in Section 5. Then, based on the control logic, the concrete traffic steering technologies such as Network Service Header (NSH) and Service Chain Header (SCH) are used to insert headers that carry service path information into packets, thus to fulfill the traffic steering process in physical network.

An example of unicast SFC is presented in Fig. 9, where four data centers offer the execution environment for VNF instances and the VNF FG is used to guide the process of service chaining. Apparently, the data centers are owned by four different service providers. Each of them provides several kinds of VNF instances. According to the request from the user U_1, it has to traverse four different VNF instances which are VNF-A, VNF-B, VNF-E and VNF-G respectively. From the perspective of VNF FG located upside Fig. 9, the red dotted line represents the logical path of traffic from U_1 to U_2. Accordingly, the logical path is mapped onto the underlying infrastructure, which is SP2 (supporting VNF-A and VNF-B), SP1 (supporting VNF-E) and SP4 (supporting VNF-G). The benefit of using VNF FG is that the service providers do not need to care about the underlying infrastructure when constructing services. That is because the VNF instances within the VNF FG have certain mapping relations with the underlying physical appliances. Interestingly, such mapping process is focused by many researchers and it is further illustrated in Section 6. In particular, these service providers are actually offering VNF instances in a pay-per-use way and they may provide the same VNFs. In this way, the users can select VNF providers according to many factors such as cost and location, which not only benefits users but also promotes competition among service providers.

- *Mobile core network virtualization*

  There are a large variety of proprietary hardware devices in current mobile network, which leads to tremendous inconvenience when any change takes place. Fortunately, NFV can reduce the network complexity and address many related issues via consolidating different proprietary network equipment into the COTS based hardware using standard virtualization technologies. In this way, various mobile network functions can be abstracted from the hardware and implemented in the form of software by using NFV technologies. Due to such decoupling, NFV offers the mobile network a complete virtual environment with a certain extent of programmability enabled. Based on this, more and more third-party innovative applications are developed to satisfy the ever increasing requirements of users, which brings many benefits to network management and operation.

  3GPP is a standardized association targeting on developing specifications and architectures for the mobile core network. Let us consider the case of Long Term Evolution (LTE) Evolved Packet Core (EPC) network, and it is composed of the following four main functions: Packet data network GateWay (PGW), Serving GateWay (SGW), Mobility Management Entity (MME) and Home Subscriber Server (HSS). Among them, the gateways (SGW and PGW) work in user plane, that is, they transport IP data traffic between user equipment and external network. In contrast, the MME works in the control plane, and it handles signaling related to mobility and security. HSS is a database that contains user-related and subscriber-related information. However, taking practical situations into consideration, we can find that not all operators have the ability to radically transform to virtual EPC from the conventional EPC once for all due to many reasons, for example, the high cost of replacing the already owned proprietary devices with COTS hardware. In this way, some operators may want to fulfill this transformation process gradually, for instance, virtualizing the HSS in the first stage and then virtualizing the P/S-GW in the second stage. In this way, EPCs belonging to different operators would have different virtualization levels [15]. In addition, the Radio Access Network (RAN) functions connect both the base station and core network elements, and the virtualization standards of RAN are not mature enough, which makes the virtualization of RAN much more difficult [174].

  Fig. 10 presents an example for LTE, in which the conventional LTE architecture and the virtualized LTE architecture are compared. In particular, the former is shown in the upside of Fig. 10, which is static and ossified due to the existence of a large amount of proprietary hardware. The latter is shown in the downside of Fig. 10, which separates the network functionalities from hardware and consolidates them in COTS hardware platform in a centralized manner. The transformation from the conventional LTE architecture to the virtualized one has many benefits. Firstly and obviously, the flexibility of service orchestrating and scheduling is improved. This advantage is explained

by literatures [13,14] and [15], among which, Akyildiz et al. [13] and Yang et al. [14] researched the 5G cellular systems based on the joint architecture of NFV and SDN, and presented the qualitative evaluation on flexibility. Hawilo et al. [15] mainly surveyed the challenges of integrating NFV with the next generation mobile network and emphasized the agility of using NFV to facilitate the virtual mobile network. Secondly, the energy efficiency is enhanced as we can easily close the unused virtual machines or re-open the required ones due to the fact that functions are deployed on virtual platforms. Along with the decoupling between hardware and network function, the underlying proprietary infrastructure can be replaced by cheap and general-purpose devices. In this way, the total CAPEX is reduced, and this kind of profit was verified by Abdelwahab et al. [27].

- *Content delivery network virtualization*
  Due to the massive and ever growing video traffic generated by end users, carrier networks are facing a lot of challenges, and the content delivery is regarded as one of them. Along with the growing video traffic, the customer's requirements on service quality are also evolving. Traditional infrastructure based networks are able to provide content delivery services demanded by customers. However, the service quality offered by traditional networks cannot be guaranteed. Therefore, the obvious contradiction between high demand and low quality leads to the declination of user experience. As a result, TOs start to integrate Content Delivery Network (CDN) cache nodes into networks, which can be a cost efficient and effective method to process such massive traffic requests with a good service quality guaranteed. Gaining the content from CDN-nodes near the customer instead of the remote source nodes can save plenty of network resources and cost, which also allows delivering data streams with high bandwidth and high quality [175].
  Unfortunately, more and more third-party entities (e.g., CDN providers) are deploying their specialized and diverse cache devices in the network, leading to the fact that the CDN network is becoming increasingly ossified and burdened. This situation has caused many issues such as wasting of resources, increased complexity and cost of carrier network, and being impossible to react to unforeseen needs.
  The CDN is composed of two parts which are distributed cache nodes and the centralized controller. In particular, the cache nodes restore some necessary content while the controller is used to determine which cache node should be selected to deliver the required content [176]. In this regard, the virtualization of CDN includes two aspects which are controller virtualization and cache node virtualization. Among them, the cache node virtualization is focused by public for the purpose of gaining preferable performance on some aspects such as network latency and throughput. For example, as shown in Fig. 11, the latency of obtaining the required content is reduced for users $U1$ and $U2$ by providing the virtualized cache nodes at NFV sites near the users. Besides, virtual network nodes can share information to realize flexible resource allocation and network load balance.

4.4.2. NFV solutions

An open source NFV ecosystem is desired by almost every service provider. However, various device vendors have different technological methods to constitute the underlying NFVI as well as some other components in NFV. In this way, the final NFV solutions offered by different service providers may vary from each other, due to the fact that service providers may select different NFVI implementations and devices from different vendors. Currently, there are a lot of tools that have introduced their specific NFV solutions, and the representative ones are illustrated as follows:

- *Global Environment for Networking Innovation (GENI)* [48]: GENI is a distributed virtual laboratory sponsored by the US national science foundation and it aims at addressing the widespread concerns of the Internet ossification which severely limits the potential for innovation. GENI is currently enabling a wide variety of experiments in a range of areas (e.g., protocol design and evaluation) and leading an effort to explore the potential of new emerged technologies such as SDN and NFV. As the SDN standard has gained substantial attention, several testbeds are developed in the GENI project. For example, the TangoGENI interconnects seven university campuses with multiple openflow based Virtual Local Area Networks (VLANs). Another testbed example is a multi-vendor openflow network testbed deployed within the SCinet research sandbox. With respect to NFV, there is no explicit work in GENI. However, GENI offers access to wide spread resources (including virtual machine and bare-machines) and supports virtualization technology. In this way, these testbeds can also be used to test NFV applications (i.e., VNFs) like virtual WAN optimization.
- *Huawei NFV OpenLab* [177]: Huawei has announced the establishment of the online NFV open lab to public in January 2015. According to the report of Huawei, they have built up a multi-vendor verification platform in NFV open lab in terms of variable and classical business scenarios. In order to provide the available data for network operators to make their decisions on NFV network programming and design, Huawei intends to construct the big data analysis platform through the continuous project integration and practice. Based on this, Huawei can cooperate with other organizations and network operators to jointly design NFV solutions. Currently, the cooperation groups include VMware and Red Hat, etc.
- *EmPOWER* [178]: EmPOWER is an open source experimental testbed which introduces the concept of NFV in the wireless network environment, thus to achieve a fully virtualized network environment for experiment. Besides, by realizing the virtual facilities, EmPOWER allows evaluating and testing the novel ideas or algorithms in large scale SDN and NFV scenarios. The EmPOWER testbed is composed of 30 nodes and used by students and research staffs in the University of Trento. In EmPOWER, the network is divided into many isolated slices. Each experiment can take full control of one or several slices such that different experiments will not affect each other. Considering the coexistence of multiple experiments, the traffic may come from different users who are joining a certain experiment or just mirroring the production traffic. Furthermore, within EmPOWER, users can also monitor the network conditions (e.g., energy consumption) in real time at either device level or slice level by using Energino toolkit [179] and Thor toolkit [180] respectively.
- *OpenSDNCore* [181]: OpenSDNCore is a software environment for verifying NFV and SDN concepts and implementing networking functionalities on top of data centers or COTS based infrastructure. From the perspective of architecture, it includes three parts which are all software implemented. The first part is an open SDN orchestrator that provides ETSI MANO aligned management and orchestration functionality, namely, managing the installation and life cycle of VNF instances, and orchestrating the VNF constituted network services. The last two parts are OpenSDNCore switch and controller that based on openflow 1.4, and the OpenSDNCore controller supports a JSON based northbound interface rather than using REST API.
- *OpenStack* [49]: OpenStack is originally an open source project for cloud computing and aims at providing a simple but scal-

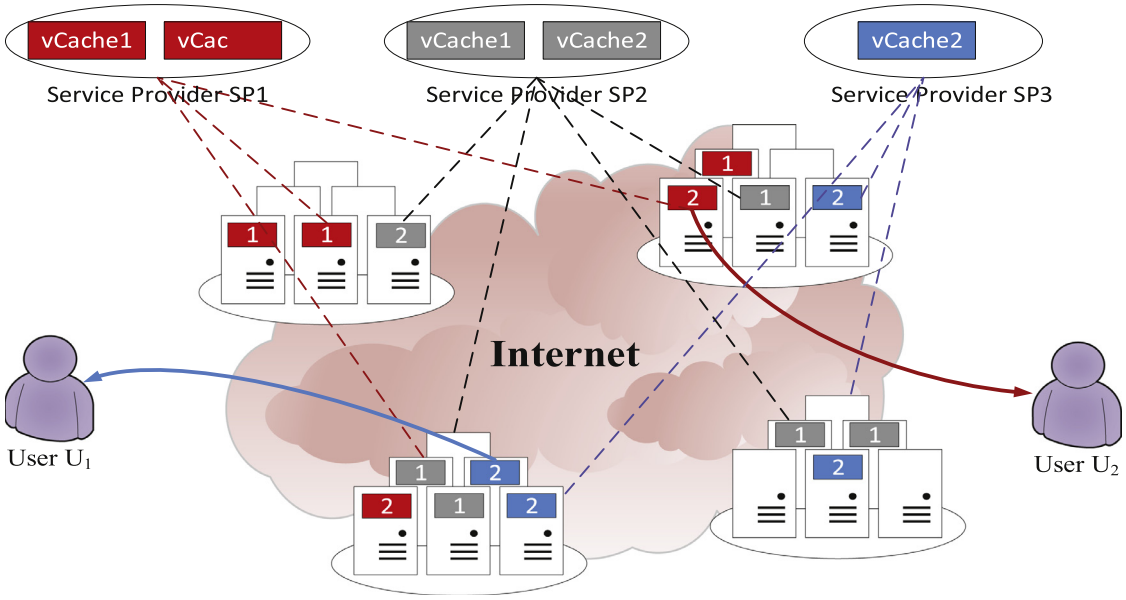

Fig. 11. The use case of content delivery network virtualization.

able and unified cloud computing management platform. The primary features of NFV are first introduced in the 10th version of OpenStack (i.e., Juno) and fully supported in the stable version Mitaka (2017). NFV is proposed to replace the legacy and proprietary hardware with high volume commercial equipment for the purpose of building a flexible service construction and provision ecosystem. Meanwhile, OpenStack intends to design an IaaS based service model which allows automated and fast VNF deployment and provision over generic hardware. Besides, in order to support large scale VNF deployment, OpenStack even provides the basis to develop essential plugins and API extensions for device vendors and service providers. Though most service providers around the world are looking for an open and multi-vendor NFV platform based on OpenStack, many aspects should be extended before OpenStack can fully realize the benefits of NFV. For example, OpenStack is currently unable to federate multiple network domains.

- *HP OpenNFV* [182]: HP OpenNFV provides an open, NFV-ready reference architecture which simplifies the transition from conventional networking to NFV-based networking. This HP OpenNFV reference architecture is carrier-grade and consists of two critical parts, that is, network infrastructure and NFV orchestrator. In particular, the former is implemented as the HP NFV System while the latter is implemented as the HP NFV Director. Although this HP OpenNFV is based on ETSI NFV architecture, it is open source to HP products and the third-party solutions. Moreover, HP has a deep alignment with SDN to better promote the development and standardization of NFV. The NFV architecture of HP declares two advantages. At first, it is layered so that multiple functionalities can be appended gradually by customers. Secondly, the customers have the ability to choose tools and equipment vendors they prefer, which reflects the openness and flexibility of HP OpenNFV.
- *CloudNFV* [20]: CloudNFV is a consolidated project proposed by multiple companies (e.g., 6wind, Dell and CIMI). CloudNFV leverages technologies of SDN and cloud computing to implement an open NFV platform in the multi-vendor network environment. Besides, it provides backward compatibility with legacy services. The architectural solution given by CloudNFV contains three parts: Virtualization, NFV orchestration and NFV management. The virtualization part abstracts network services, functions and resources. The orchestration part fulfills the VNF deployment and service orchestration according to the required VNF order and network resource status. Services are delivered when traffic passes through these VNF, and NFV management controls all the VNF-related actions including initiating, monitoring, and destroying VNF instances, etc. CloudNFV considers management and orchestration systems as applications, which is different from ETSI NFV MANO.
- *OPNFV* [52]: OPNFV is a carrier-grade, integrated and open source platform to accelerate the introduction of new NFV products and services. OPNFV is uniquely positioned to bring together the work of different SDOs, open source communities and commercial suppliers for the purpose of delivering a de facto standard open source NFV platform. By integrating these works from different organizations, OPNFV builds the NFVI and VIM. Such two components, along with the API and other auxiliary NFV elements, form the basic framework of NFV. Importantly, since the NFV solution offered by OPNFV is composed of the work from different organizations, its portability and suitability in terms of various NFV use cases can be guaranteed.

4.4.3. Coexistence with legacy systems

The coexistence between NFV and other legacy systems (e.g., Business Support System (BSS) and Operational Support System (OSS)) is inevitable [183]. The operations applicable for legacy systems might not work for NFV, and vice versa. For example, the static network configuration tasks managed by BSS/OSS should be separated from the dynamic real-time management of network states managed by the SDN controller. Besides, the service in legacy system is only composed of PNFs, while the service in the NFV and legacy system co-existence environment is usually provisioned by the hybrid of PNFs and VNFs. However, considering the characteristics of PNFs, the VNFs are generally deployed to accommodate the existing PNFs for service provision. This actually limits the flexibility of service provision and may even cause performance degradation when specific VNFs are not placed in the optimal positions [184]. In particular, once a problem occurs for the service provision, it is usually due to the mistakes made in the initial configuration or changes made to the environment.

With the exhaustion of IPv4 addresses, IPv6 is playing an important role. To enable IPv6 as a fundamental feature inside the

NFV architecture, all the NFV components (e.g., MANO, VIM and NFVI) should take specific considerations to perform IPv6 supported functions and maintain the backward compatibility with IPv4 at the same time. Besides, given the large address space offered by IPv6, some mechanisms (e.g., the floating IP) used to economize the IP addresses in IPv4 may not be required as necessary, and this could relieve the burden when applying NFV into IP networks [2]. The IP multicast is another important issue, which can actually be regarded as a distributed routing problem and it mainly focuses on constructing the multicast topology. However, for the multicast in the context of NFV, it requires not only constructing the multicast topology, but also deploying the required functions (or VNFs) in the network, so as to steer traffic through these functions before reaching destinations [185]. Both the multicast topology construction and VNF deployment problems are NP-hard. Therefore, the NFV enabled multicast problem is much more complex than the traditional IP multicast problem. Reviewing the related work of NFV multicast, it is discovered that most of them deviated from the original intention of NFV multicast. Specifically, they directly regarded the NFV multicast problem as a Virtual Network Embedding (VNE) problem, and addressed the corresponding VNE problem. However, the VNE problem is to embed the virtual network requests onto infrastructure, while the NFV multicast problem needs to not only embed VNFs onto infrastructure, but also steer traffic through these VNFs in order. Despite the similarities between such two problems, they are actually different.

There are many other issues that must be addressed before embracing the benefits of NFV, for example, the compatibility between NFV and legacy systems, the requirement for an automated resource and VNF life cycle management system, and the lack of suitable integration methods with cloud models, etc. Anyhow, there are two ways to address these gaps: either by modifying the legacy systems or by adopting NFV augmentation tactics. Specifically, the former indicates that many aspects (e.g., data model, processing engine and compliant interface) of the legacy systems should be modified [25]. However, such modification requires a relatively long period and high cost. The latter indicates that many service management modules should be integrated in NFV in order to provide an overall service orchestration for legacy systems by abstracting both physical and virtual network elements [186]. Comparatively, the latter does not need to change the legacy systems or a long time for implementation. However, the service management modules must be designed to fully augment the legacy system capabilities, otherwise, the effect may not be obtained as expected.

From Sections 4.1 to 4.4, the concept of NFV is comprehensively introduced in a bottom-up way. With respect to each layer, we also review the corresponding tools for NFV. Although they are explained in a specific hierarchical structure, it will be much easy to understand NFV with a global knowledge view. Thus, we organize the key points of NFV and summarize them in a diagram, which aims at presenting a tutorial and vivid view of NFV as shown in Fig. 12.

5. VNF related algorithms

Due to the decoupling of network functions from the dedicated hardware, the efforts are shifting from hardware to software. Among these efforts, the algorithms play an important role. Considering the importance of VNFs, we comprehensively investigate and illustrate the VNF related algorithms in this section.

5.1. VNF placement

In NFV networks, network functions are decoupled from the underlying hardware and implemented as VNFs. Due to the software-feature of VNFs, they can be flexibly deployed. Therefore, a critical problem appears, that is, how to determine the positions for placing VNFs such that the service requirement and quality can be satisfied. Such problem is referred to as the VNF Placement (VNF-P) problem which is proved to be NP-hard [187]. In this regard, it is usually hard to find the optimal solution for VNF-P especially in large scale network scenarios.

In order to achieve the optimal solution for VNF-P, the mathematical programming methods such as Integer Linear Programming (ILP) and Mixed ILP (MILP) are generally used. For example, Bari et al. [187] introduced the VNF orchestration problem which was equal to VNF-P, and formulated it as an ILP model in terms of minimizing the OPEX and maximizing the network utilization. Riggio et al. [188] formulated the VNF-P in radio access network scenario as an ILP model which was solved to achieve the optimal VNF placement solution under the radio resource constraints. Besides, other different objectives were also considered when solving VNF-P, for example, Luizelli et al. [189] formulated the ILP model targeting on minimizing end-to-end delay and resource overprovision ratio, while the objective of Gupta et al. [190] was minimizing the bandwidth consumption. Although they had different objectives and worked in different scenarios, the constraints of VNF-P were generally the same, which included those on resource allocation, VNF mapping and Traffic Engineering (TE) (e.g., quality of services).

However, ILP is only suitable for the situation that all variables are integers. Thus, for some specific situations, the MILP is used instead. For example, Addis et al. [191] proposed a VNF-P model, in which both the NFV goal (minimizing the number of CPUs used by instantiating VNFs) and the TE goal (minimizing the risk of sudden bottleneck on network links) are considered. However, in order to jointly achieve the two goals, some non-integer variables have to be introduced. Thus, the VNF-P model proposed by Addis et al. [191] was actually a MILP model which described the relationship between VNF placement and traditional routing. By solving this model, Addis et al. [191] claimed to achieve a 70% cost saving on NFVI and a 5% increment on link utilization compared with those only considering the TE goal.

The ILP and MILP models are usually solved by the open source optimization software (e.g., CPLEX [192], LINGO [193] and GLPK [194]) to achieve the optimal solution. Other classical algorithms solving the two mathematical models include branch-and-bound, branch-and-cut, etc. Nevertheless, these mathematical proposals suffer from an obvious scalability weakness, that is, they are not applicable in large scale networks, because their execution time grows exponentially with the network size. For example, Bari et al. [187] spent 1595.12 s using CPLEX to solve the ILP model in terms of a network scenario with only 23 nodes and 43 links. Fortunately, most of the literatures mentioned above were aware of this problem and they proposed the corresponding heuristic algorithms right after solving their models with the optimization software or exact algorithms. For example, with regard to the same scenario (i.e., 23 nodes and 43 links), Bari, et al. [187] proposed a multi-stage graph based heuristic which solved VNF-P using 0.442 s. Another selling point of Bari et al. [187] was that it targeted on minimizing the physical resource fragments by frequently measuring the percentage of idle resources on the active servers or links. Due to such consideration, Bari et al. [187] claimed a 4 times of OPEX reduction compared to the middle-box based solution, and up to a 1.3 times of OPEX reduction compared to the optimal solution. It is known that the NFV based network will coexist with the legacy networks for a long time. However, Bari et al. [187] was proposed in terms of the former. Thus, it was hard to evaluate whether this work was suitable for practical situations.

Due to the fact that the execution time of heuristic is much lower than the optimal solution and most of current heuristics can provide the results approaching those achieved by optimal solu-

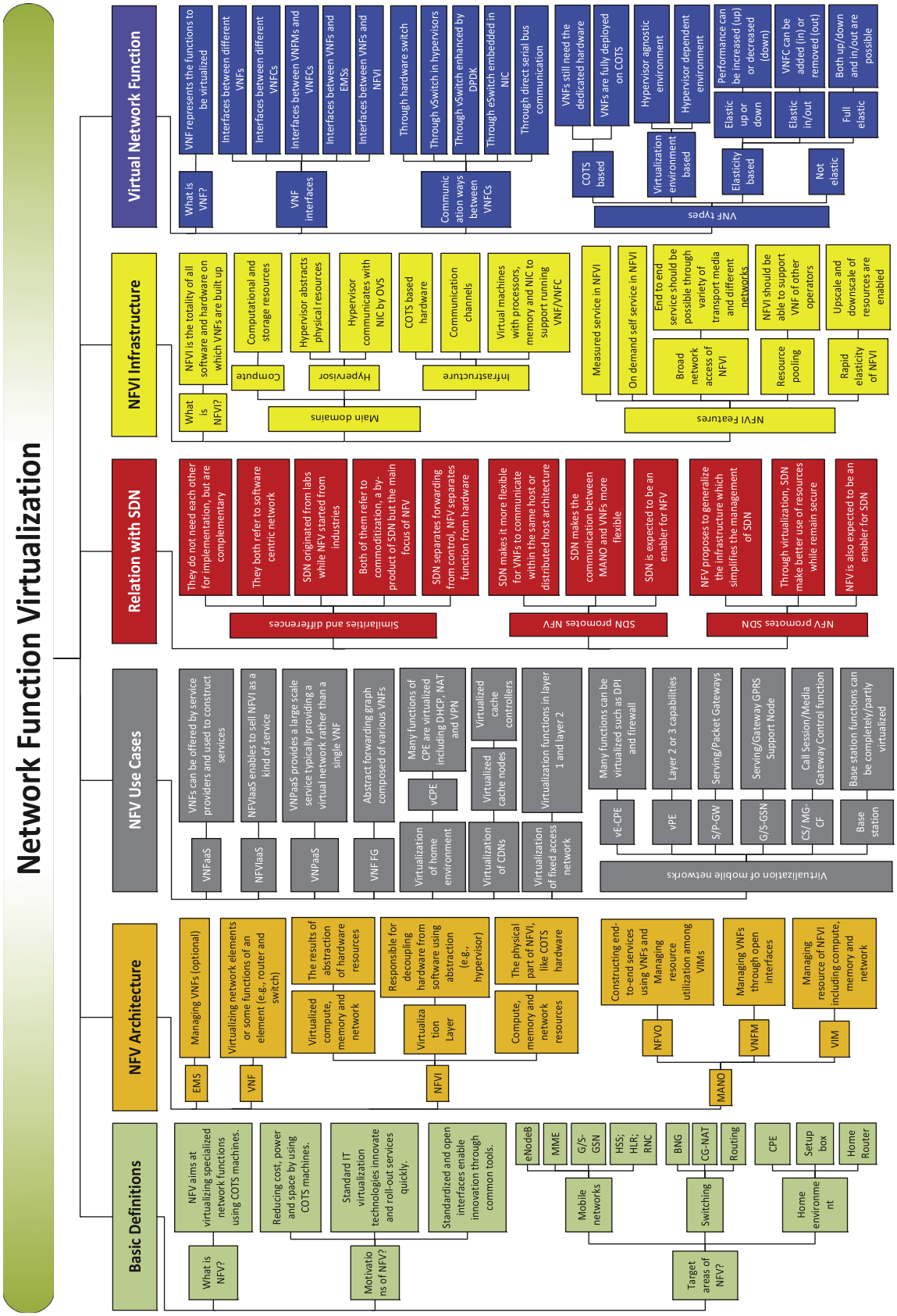

Fig. 12. A tutorial diagram for NFV.

tion, more and more heuristics are proposed to solve the VNF-P in terms of different network scenarios and metrics. For example, for VNF-P in wireless networks, Riggio et al. [188] proposed a three-step heuristic algorithm, that is, computing candidate substrate nodes for virtual nodes, sorting virtual nodes and mapping them onto substrate nodes. Following the three steps, Riggio et al. [188] claimed an approximate performance to the optimal one. However, this work actually converted the VNF-P to a VNE problem for solving. Despite the similarities between VNF-P and VNE, they are in fact different problems.

Addis et al. [191] introduced the concept of legacy TE into NFV for the purpose of achieving both the TE goal and NFV goal. In particular, the TE goal was to minimize the maximum network link utilization, while the NFV goal was to minimize the number of CPUs used by VNF instantiation. However, the two objectives were in competition, because minimizing the maximum link utilization required to deploy extra VNFs for the load balancing purpose. In order to address such contradiction, the two objectives were prioritized first. Then, the best solution of the first priority objective was calculated, which was regarded as the input of the second priority objective. The second objective might not be fulfilled as desired, because the corresponding solution was to iteratively increase the value of the first objective until the desired metric (e.g., cost) of the second objective was satisfied. Similarly, Luizelli et al. [189] iteratively searched the solutions for the VNF placement problem. In particular, Luizelli, et al. [189] proposed to solve the VNF-P in a way that the objective of the previous iteration was transformed into a constraint of the next iteration. In this way, Luizelli, et al. [189] claimed to achieve a more fine-grained and effective solution. In particular, the binary search scheme was used in order to quickly find the lowest possible number of VNFs that met users' requirements. Nevertheless, this proposed heuristic algorithm was still an exact approach, which meant a comparatively high time complexity. However, the above work did not take the existence of PNF and VNF into consideration, and were hard to be used in the practical situation. In this regard, Moens, and Turck [195] proposed to solve the VNF-P problem under the hybrid environment, that is, the network services were constituted by both PNFs and VNFs.

Khebbache et al. [196] proposed to solve the VNF-P based on the concept of a multi-stage graph. In particular, it first constructed a multi-stage graph representing servers available to host the required VNFs. Then, the service chains corresponding to the VNF forwarding graphs were fulfilled using a maximum flow between the vertices of different stages of the multi-stage graph representation. Nevertheless, the major problem of this work was that it limited the number of VNFs in a service chain to 3, which did not satisfy the practical situation. Pham et al. [197] proposed an algorithm based on Markov approximation technique to solve the VNF placement problem. Specifically, it started with an arbitrary chosen feasible VNF placement solution, and might move to another feasible one according to the network states. This approach converged when the Markov chain reached to the stead-state distribution. However, it required a long convergence time to find the near-optimal solution, which was due to the large state space of this optimization problem that depended on the combination between chosen subsets of physical nodes and VNF placement schemes.

5.2. VNF scheduling

The VNF Scheduling (VNF-S) problem is different from the VNF-P problem and VNF-S generally has two research points. In particular, the first research point is to schedule different VNF instances to serve and provide network services. Under this situation, the VNF-S problem is usually processed together with the VNF-P problem since they have the same goal, i.e., service provision. For example, Riggio et al. [188] first formalized the VNF-P problem for radio access network. Then, based on the formulation, it proposed a slice scheduling mechanism for VNFs, which guaranteed resource and performance isolation between different slices. The performance isolation was achieved if the resource slices were accepted under the constraints imposed by the proposed VNF-P problem formulation. Comparatively, Lucrezia et al. [198] introduced a network-aware scheduler, that is, the scheduler had the ability to optimize the VNF placement from a networking perspective, which was essential for deploying VNF service graphs efficiently. Such scheduler was implemented under the environment of OpenStack and it also extended OpenStack with traffic steering primitives. It was used to instruct the traffic to follow arbitrary paths among VNFs and to allow packets to traverse VNFs without the need of explicit routing rules.

Another example that VNF-S and VNF-P are jointly solved can be found in Ref. [199] which proposed three heuristic algorithms and one metaheuristic algorithm. Specifically, the three heuristics were all based on greedy strategy but with different criteria, and the metaheuristic was based on tabu search (a search method based on local search). With respect to the VNF mapping, the first one intended to map VNFs on the nodes with best processing time, the second one mapped VNFs on the nodes with the queue of the earliest completion time, and the third one mapped VNFs on the nodes with the highest available buffer capacity. After that, the VNFs were scheduled using the shortest path in order to provide the required services. However, the three greedy algorithms mapped and scheduled the VNFs sequentially, which was inefficient. Meanwhile, the tabu search based algorithm leveraged local search mechanisms for mathematical optimization and its objective was to minimize the flow scheduling time. Besides, based on the characteristics of tabu search, the proposed metaheuristics was fulfilled in five components, that is, the initial solution for VNF mapping and scheduling, the neighborhood solution (similar to the initial solution), tabu list (a set of candidate nodes for VNF mapping), aspiration criterion (the criterion that allows moving to another solution) and stopping condition.

As explained, the above work targeted on scheduling VNFs for service provision. However, following Brucker's guidelines, a scheduling problem should find the time slots in which different activities can be processed under given constraints [200] (e.g., resource constraints). Therefore, the second research point of VNF-S problem is to find the corresponding time slots for flows on the set of VNFs to traverse. In order to solve such problem, Riera et al. [201] introduced a two-stage method, which at first mapped the VNFs onto corresponding servers, and then the VNF-S problem was formulated as a resource constrained scheduling problem, in which the objective was to determine a feasible schedule with minimum makespan. In addition, such two stages were implemented within the SDN paradigm. Similarly, Shifrin et al. [202] solved the VNF-S problem under the SDN environment. It presented two models for this problem, that is, the queueing system model in which the number of queues was flexible and a variety of constraints were considered (e.g., delay and cost), and the Markov decision process model in which the optimal policies were made for VNF scheduling. In spite of the two optimization models, Shifrin et al. [202] did not give an efficient method for solving VNF-S problem, which indicated inapplicability for large scale network scenarios.

Qu et al. [203] and [204], published by the same authors, were proposed to formulate the VNF-S problem as a series of scheduling decisions which activated various VNFs to process the traffic of different services. Based on the defined integer and non-integer variables, the formulation was managed in a more fine-grained manner, in which minimizing the latency of VNF scheduling was focused. The lower the latency, the more customers are served, which naturally leads to more profits. Qu et al. [203] mainly focused on how to assign the execution time slots to different ser-

vices traversing the same VNF via using a greedy method, while Qu et al. [204] considered not only this aspect, but also the resource allocation. In particular, Qu et al. [204] decomposed the proposed VNF-S model into four sub-problems, that is, *i*) the virtual link bandwidth allocation sub-problem (assigning each virtual link a feasible data rate); *ii*) the VNF assignment sub-problem (allocating VNFs of each network service to VMs); *iii*) computing the transmit rates at the VNFs of each network service; *iv*) generating a feasible schedule for each VNF via a greedy method. Based on such decomposition and the Genetic Algorithm (GA), Qu et al. [204] developed a simplified heuristic method for solving the VNF-S problem with transmission delay considered. Therefore, by dynamically adjusting the allocated resources, Qu et al. [204] reduced the scheduling time by 15–20%, while Qu et al. [203] saved roughly 14.3% scheduling time.

5.3. VNF migration

The VNF Migration (VNF-M) problem generally refers to the process of migrating VNFs from one place to another due to specific requirements such as load balance and hardware maintenance [205]. During the process of migration, the VNF related state (e.g., CPU interrupt and memory) must be migrated to the destination as well. However, the new VNF may ignore some malicious activities due to the lack of necessary information [206]. In addition, VNFs are actually software that can be implemented in either VM (the isolated duplicate of a real computer machine provided by KVM, VMware, etc.) or container (a stand-alone and executable environment for software, e.g., Docker). Typically, VM is used to host VNFs in the research work [207].

The VNF-M problem is evolved from the traditional VM migration problem which includes two kinds of migration situations. The first one is cold migration [208], that is, the power of VMs must be shut down before migration and it is usually applied to the disk data migration. The second one is live migration [208] which refers to the process of migrating a running VM between different physical machines without disconnecting the client or application, until the corresponding information of the VM are transferred from the original guest machine to the destination. Considering their features and users' requirements, the live migration may occur with a big probability in real world environment. Typically, there are three techniques for live migration, which are pre-copy, post-copy and the hybrid of them [51]. In particular, for post-copy migration, it first sends the processor state to the destination, and then transfers the VM's memory contents. However, the pre-copy migration does the opposite, that is, the pre-copy migration first repetitively copies the memory state to the destination and then transfers the processor state [207]. Although these strategies can also be applied to solve the VNF-M problem, specific VNF features must be taken into consideration. For example, traditional VM migration includes cold migration and live migration [208], however, the VNF-M is usually processed as a live migration problem due to the real-time feature of many VNFs.

The VNF-M problem exists in many network scenarios. For example, with respect to the NFV networks, Eramo et al. [209] and [210] were proposed to deal with the VNF migration problem, during which the energy consumption was saved. In particular, the former proposed a heuristic method based on the Viterbi algorithm [211] (a kind of dynamic programming algorithm) to determine the migration policy, that is, when and where to migrate the VNF instance in response to the changes of service requests. However, the migration technique used by the former was actually belonging to the cold migration in which the virtual machines were redundant and suspended before migration. With respect to the latter (i.e., Ref. [210]), it also leveraged the Viterbi algorithm to determine the migration policy. Nevertheless, unlike the former that only focused on minimizing the total energy consumption in terms of VNF consolidation and migration, the latter focused on VNF placement, service routing and VNF instance migration in response to the changing workload. Thus, the technique it adopted was live migration.

With respect to legacy environments, e.g., the Fixed Mobile Convergence (FMC) network, Andrus et al. [212] implemented two mechanisms for solving the VNF-M problem. The first one was to build a dedicated connection for VNF migration. Along the dedicated connection, the VNF was migrated without breaking off the service. Thus it yielded a zero service downtime and was regarded as the benchmark in [212]. With respect to the second one, it targeted on optimizing the network utilization by migrating the VNFs such as content caching and performance monitoring to servers closer to the client devices. The migration process was implemented by the SDN controller which offered a centralized view to help build the path between migration source and destination, such that the VNF migration can be fulfilled. Nevertheless, we should be aware that achieving the zero service downtime for the first mechanism was actually costly, which might not be accepted by enterprises or individuals.

The live migration of VNF is a common topic in cloud based networks. Under this situation, Cerroni, and Callegati [213] presented a migration model for multiple VNFs, which was used to derive some performance indicators such as the service down time and the total migration time. Besides, Cerroni, and Callegati [213] also proposed two schemes for migrating multiple VNFs. The first one was the sequential migration which migrated one VNF at a time, and the second one was the parallel migration which migrated multiple VNFs simultaneously. Apparently, the sequential migration was easy to be implemented but inefficient when it required migrating a large number of VNFs. Meanwhile, the parallel migration was complex but it migrated multiple VNFs efficiently. Thus, the problem of Cerroni, and Callegati [213] was in fact to make the trade-off between the two schemes. Nevertheless, the pre-copy migration strategies were adopted by the two schemes, that is, the whole memory of the source VNF was copied to the destination while it was still running at the source node. In addition, with respect to the environment of virtual content distribution network, Hatem, et al. [214] specifically focused on the migration of the virtual content delivery functions. In particular, the flow balance and conservation were considered when determining the optimal destinations for placing virtual content delivery functions. Although it claimed to address the problem of migrating the virtual content delivery functions with an intelligent algorithm, it actually formulated this problem as an ILP model and solved it using the CPLEX tool.

The live migration of VNF related state is very important for solving the VNF-M problem. However, such process introduces performance degradation, for example, the increased jitter and packet loss. To minimize the impact, many efforts have been proposed to reduce the migration time, thus to reduce the performance degradation as much as possible. One typical example was shown in Ref. [215] which formulated the VNF-M problem as a MILP model and proposed a method based on linear approximation and fully polynomial time approximation to solve it. For each migration, it first computed the optimal migration sequence and the required network bandwidth. Based on the sequence information, the VNF states were migrated and the calculated bandwidth was used to guarantee the quality of migration. Compared with the traditional VM migration algorithms [216,217], the proposed method claimed to save the migration time and service downtime by up to 40% and 20% respectively. In addition, to enable an efficient and seamless VNF state migration, Nobach et al. [206] intended to provide VNFs with interfaces which were used to announce the changing states for incoming packets, such that the VNF state synchronization could be obtained efficiently with a low cost. Although it

claimed to achieve high physical link utilization (around 3 times than some existing approaches) for the seamless VNF state migration, these interfaces would inevitably introduce issues such as security and management.

Currently, there appear many automatic VNF migration frameworks (such as Split/Merge [218] and OpenNF [219]), they are typically designed in terms of the SDN environment and used to fulfill the migration of traffic flows and their states. However, such operation required hundreds of milliseconds to complete, and it usually generated a lot of overhead in the control plane and degraded the application performance. For example, it took OpenNF more than 100 ms to move per-flow states for 1000 flows, which significantly increased the flow completion time of mice flows. In addition, moving all flows required the controller to update many flow entries in the routing tables, causing significant overhead at both controller and switches with limited flow table size. Instead of migrating all the states, Liu et al. [220] proposed a novel algorithm which only migrated the states related to elephant flows, because the short-lived mice flows generally expired before the migration ends. Hence, there was no need to migrate them. Compared with OpenNF, it reduced the migration time and latency by 87% and 94% respectively. Furthermore, Wang et al. [221] decoupled the processes of state transfer and data packet migration, such that they could be executed and optimized in parallel. For instance, the incoming packets of migrating flows could be redirected when the migration started without waiting for the finish of state transfer. Although such design resulted in 3 times shorter migration time than the existing approach [222], it inevitably imposed the overhead for controlling the separated data packet migration, and incurred the state synchronization problem.

5.4. VNF chaining

The VNF Chaining (VNF-C) problem is also called service function chaining problem which mainly focuses on the mechanisms for chaining VNFs and steering the corresponding traffic through these VNFs in order before reaching destinations [223]. Targeting on this problem, IETF has specially established a working group to document new approaches for VNF constituted service delivery and operation as well as the SFC architecture and algorithms for traffic steering [68], which proves the importance of SFC. In order to keep a unified style, the term VNF-C is used instead of service function chaining in this section.

In order to address the VNF-C problem, Sahhaf et al. [184] and Lee et al. [224] formulated it as an ILP model with different objectives. The former intended to minimize the total cost by making a reasonable selection of the VNF decomposition, while the latter intended to minimize the end-to-end latency with random flow distribution. With respect to the former, it formulated the VNF-C problem as an ILP model which was solved by COIN-OR [225]. Besides, it also proposed a heuristic algorithm to solve the VNF-C issue, which could be divided into two parts, that is, decomposition selection and mapping. The decomposition selection part was to prioritize the decomposition of VNFs while the second part was to properly map VNFs on substrate nodes based on the backtracking strategy. With respect to the latter, it first introduced two benchmarks which were uniform distribution scheme (flows were evenly distributed to all VNF instances and no resource constraints were considered) and network-aware distribution scheme (flows were distributed based on the latency between any two VNF instances) respectively. Then, by changing the flow distribution, it claimed to reduce the total latency by 27.14–40.56% and 12.77–28.84% respectively when compared to the two benchmarks. Nevertheless, no explicit explanations were given.

Apart from solving the VNF-C problem in a static manner, many other works intend to study this problem in a dynamic way, which is more suitable for practical situations. In this respect, Liu et al. [226] considered the re-adjustment problem of VNF-C in a dynamic environment. Specifically, it had to fulfill the new service requests on one hand. On the other hand, it also had to re-adjust existing service chains in order to satisfy the changing requirements of users. In order to solve the mentioned two situations, Liu et al. [226] first built the corresponding ILP model which could be used to obtain the optimal solution. However, due to the extremely high execution time spent on solving the ILP model, Liu et al. [226] also proposed a heuristic method based on the idea of Column Generation (CG), which only needed to generate the variables that might improve the objective formulated in the ILP model. Besides, the main idea behind the CG based algorithm was to decompose the original problem into a master problem and a sub-problem, and then solve them iteratively in order to obtain the near-optimal solution.

Due to resilience and economic reasons, many service providers would like to consider the VNF-C problem and the VNF placement. In order to achieve such joint objective, Bouten et al. [227] proposed a model for VNF-C, in which not only the resource capacity constraints, but also VNF location constraints, were considered. Similarly, the corresponding proposed heuristic algorithm was also implemented under such constraints. Nevertheless, it was discovered that this proposed heuristic regarded the VNF-C problem as the VNE problem, and thus the problem actually solved was VNE instead of VNF-C. Although the VNF-C problem looks similar to the VNE problem that has already been studied intensively (e.g., [228] and [229]), they are fundamentally different. For example, for VNE, the topology of virtual network requests does not change during the embedding process. However, for VNF-C, the topology of service requests may change according to the deployment of VNFs.

Most of the work mentioned above studied the VNF-C problem in the context of SDN, that is, the VNF-C problem was usually solved in a centralized manner. However, distributed solutions for VNF-C were also required and DOro et al. [232] implemented a distributed service chaining. Specifically, the proposed solution of D'ro et al. [232] was based on the non-cooperative game theory which indicated the game with competition between individual players and only self-enforcing alliances were possible in the game. In addition, to account for selfish and competitive behavior of customers, the service chain composition was formulated as an atomic weighted congestion game which processed a weighted potential function and admitted a Nash equilibrium (a state that no player could benefit by changing only his own strategy). Nevertheless, it is worth mentioning that the requirements for services may change over time. Specifically, VNFs may be added into or deleted from service chains according to dynamic requirements. In this way, the procedures for service recomposition, function re-mapping and rescheduling are essentially required, which lack enough research by far. One example could be found in Ref. [233] which proposed to solve the scalability problem of VNF-C, that is, the VNFs required by any service may change (e.g., adding a new VNF or removing existing VNFs). It proposed two heuristics to address such case. The first one added or removed VNFs based on the reserved service function path, while the second one intended to optimize the reserved service function path according to current network state. Despite the two proposed heuristics, the joint usage of them could generate better results than using each of them exclusively.

Some literatures prefer to transform the VNF-C problem into other types of problems for solving. For instance, Li et al. [234] abstracted the VNF chaining and service path selection problem as the grey theory problem. Based on the concept of grey system theory, the grey relational grade was adopted to measure the relationship among the candidate composition of services, the ideal service composition and the negative ideal service composition.

Then, a membership function was used to calculate the membership of candidate composition in terms of the ideal service composition. Among them, the best solution was selected finally. Although Li et al. [234] claimed to achieve 200 ms average delay and 8% lower packet loss than using the random method, it calculated all the possible paths each time when a request arrived in order to choose the best one, which resulted in massive calculation burden. Meanwhile, Wang et al. [9] transformed the VNF-C problem into a Markov chain model. Then, based on existing Markov approximation method [235], it proposed another distributed algorithm in order to obtain optimum-approaching solutions. Nam et al. [236] converted the VNF-C problem to a VNF cluster and allocated problem. In particular, it first clustered the VNFs according to their popularity, and then allocated them to service requests according to their requirements.

The VNF-C problem has also been studied in many other scenarios such as cloud [237,238], datacenter [239], carrier grade network [8], etc. Other than focusing on the algorithmic aspect, these works prefer the design of service chaining architecture, for example, the framework for service chain provision and the coordination among VNFs from different vendors. In addition, according to the strategies adopted by the above mentioned algorithms, the VNF based services are usually implemented in two ways. In particular, the first one relies on using the centralized orchestrator to judge where to forward the incoming packet. Obviously, this mechanism is computing-consuming and time-consuming, because the orchestrator needs to make the judgement each time a packet arrives. Comparatively, the second one is much more efficient by leveraging the NSH encapsulation [110]. In particular, considering one service with the service path given, such path information is encapsulated in the header of its packets. Then, the packets belonging to this service are forwarded according to the header information during the whole transmission process.

Despite the fact that the above mentioned algorithms provided a lot of high-level ideas for VNF chaining and traffic steering, most of them were actually determining the routing logic of traffic among different VNFs, which could not be directly applied to practical situations. In fact, there are two primary kinds of practical technologies used for the VNF chaining and traffic steering of SFC, which are header based ones and tag based ones. Such two technologies try to insert headers or tags that carry the information of service path and functions into packets. In this way, the packets can be forwarded and processed accordingly. However, we should be aware that the content of the headers and tags are actually determined based on the routing logic achieved by the above mentioned high-level algorithms. Nevertheless, the header based technologies generally define the header format, while the tag based technologies encode tags into the available fields of packet headers. Thus, the two kinds of traffic steering technologies are different in their control granularity.

On one hand, with respect to the header based traffic steering technologies, many of them are proposed, for example, NSH [110], Service Chain Header (SCH) [240], SFC Mapping Header (SMH) [241], and Segment Routing Header (SRH) [242]. For NSH [110], it is one SFC protocol that is composed of the following three fields: service function path identification, indication of location within a service function path and an optional metadata. The first one is used for packet forwarding and the second one is used for function delivery, while the last one is reserved for purposes like policy enforcement. NSH imposes a network service header on the original packets or frames to realize the service function path. Thus, it can offer the ability to monitor and troubleshoot one SFC. In addition, some other tools (e.g., a traffic analyzer) can also verify the details of one SFC in terms of forwarding path and function chaining by using the three fields of NSH. For SCH [240], it is actually similar to NSH. The difference is that SCH only includes two fields which are a fixed length mandatory one and an optional one. The mandatory field carries the SFC path information which is used to steer the packets through an ordered set of VNFs. This function is fulfilled by the first two fields of NSH. The optional field of SCH is similar to that of NSH. For SMH [241], rather than realizing the service function path in the packet header, it focuses on identifying the packets returned from legacy service functions, which do not explicitly carry the SFC header. In this way, the legacy service functions can join in the service function chains without supporting the SFC headers. Likewise, SRH [242] is actually a routing protocol which can be used to realize the process of service chaining. Instead of inserting service function path into the packet header directly, SRH defines a new concept called segment which represents the control instruction used for packet steering. By prepending the segment routing header to the packet, SRH allows enforcing a flow through any path, while maintaining per-flow state only at the ingress node to this segment routing domain. In addition, SRH can also be applied to IPv6 with the addition of a new type of routing extension header.

On the other hand, there are also many tag based traffic steering technologies. As explained, the tag based methods usually rely on encoding the specific tags into the available fields which are usually defined by most protocols for optional matedata. Meanwhile, these tags are usually used to indicate the service function path of SFC. Jointly taking the two aspects into consideration, the tags can be determined by identifications or labels (e.g., MAC address or VLAN IDs). With respect to the MAC address, it is used in [243] which encapsulates the service chain identifier based on source MAC to achieve scalability. Specifically, for any incoming packet, the middle-boxes or switches will first parse the corresponding SFC identifier, and then add the SFC identifier with new MAC address before forwarding it out. Therefore, compared with the header based methods (e.g., NSH [110]) that only generate the SFC identifiers, the tag method proposed in [243] is both generator and consumer of the SFC identifier. Another similar work is FlowTag [244] which proposes an extended SDN architecture. This extended architecture enables middle-boxes with the ability to add tags to outgoing packets, which can be used to provide necessary context such as source hosts and internal cache states. The difference is that FlowTag bases on VLAN IDs to encode the tags. Nevertheless, we discover that the methods base on MAC and VLAN ID to describe the SFCs in the same layer. Thus, for the SFCs across multiple layers (e.g., network layer and data link layer), these methods may not be able to work in good condition. In this regard, the label technology, e.g., Multi-Protocol Label Switching (MPLS) [245], can be used. MPLS defines fixed length labels for different network domains. Once one SFC packet enters a domain, it will be encoded with a label which is used for the following switching and forwarding. In particular, these labels are used to build the Labeled Switched Path (LSP) which provides the same functionality as the service function path of SFC [246].

5.5. VNF multicast

Multicast is very important for network communication due to its advantage in bandwidth saving. However, it suffers from many limitations. Traditionally, creating and maintaining multicast trees require both high time and economy cost. Besides, changing the network functions that are already in the traditional multicast tree requires a lot of reconfiguration work, which is costly and time-consuming [247]. Unlike traditional multicast that mainly focuses on determining the route from one source node to multiple destination nodes, the NFV-enabled MultiCast (short for VNF-MC) requires to do more, that is, creating the multicast tree, placing the required VNFs on substrate nodes and steering traffic through these VNFs before reaching destinations [248]. However, NFV is a

new paradigm which enables to implement the multicast services efficiently by replacing the proprietary hardware with general-purpose hardware, thus to provide an environment for fast VNF deployment and execution.

Typically, constructing a multicast tree with minimum cost can be formulated as the Steiner tree problem which has been proved to be NP-hard [249]. Fortunately, there are already many algorithms proposed to solve this problem. For example, Vrontis et al. [249] constructed a DiffServ (i.e., a scalable mechanism for managing traffic and providing service quality for networks) based and customized Steiner tree using the shortest path algorithm and Byrka et al. [250] proposed an ILP based approximation algorithm using linear relaxation (turning a difficult problem to an approximate problem that is easier to solve) and iterative randomized rounding techniques.

However, we should be aware that only constructing a multicast tree cannot completely solve the VNF-MC problem. Therefore, based on existing efforts, many researches toward solving VNF-MC are proposed. One example can be found in Ref. [251]. It proposed a heuristic algorithm which leveraged both the characteristics of GA and Simulated Annealing (SA) strategies. On one hand, it encoded the node/link mapping, multicast topology construction and the spectrum requirements in the same gene. On the other hand, it used the SA to find the most fitting VNF mapping sequence for the arriving multicast requests. It obtained a 31–56% less resource consumption and a 4–24% higher throughput than the greedy based algorithm. In addition, the idea of GA was also adopted by Ref. [252] to address the VNF-MC problem. Specifically, Gao et al. [252] first formulated the VNF-MC problem as a MILP model based on the concept of max–min fairness in order to achieve the upper bound of service reliability which could be used to evaluate the performance of the proposed heuristic. Then, the proposed heuristic method relied on using GA to encode the multicast tree construction and link mapping into path selection. Due to the joint consideration of the max-min reliability fairness goal and the networking reliability factor, the proposed heuristic method was demonstrated to achieve near-optimal results (merely around 0.2–0.4% gap).

Nevertheless, the above work simply regarded the VNF-MC problem as the VNE problem, by solving which the solution was obtained. Despite the similarities between the VNF-MC problem and VNE problem, they are fundamentally different and they should be addressed differently. In this regard, Zeng et al. [253] explicitly explained their fundamental differences and treated VNF-MC problem differently. Specifically, it divided the VNF-MC into offline and online cases and proposed corresponding heuristics to solve them respectively. For the offline VNF-MC, it was formulated as a MILP model in the context of inter-datacenter optical network. To solve the offline VNF-MC efficiently, a heuristic method based on path-intersection was proposed, that is, it calculated all the intersection nodes of the paths from source to destinations and tried to place VNFs on these intersection nodes. However, the offline VNF-MC assumed that all the multicast requests were known in advance, which did not satisfy the practical situation. Likewise, based on the concept of path-intersection, two heuristics were proposed by Zeng et al. [253] to solve the online VNF-MC problem, that is, the batch scheme that provisioned multicast requests simultaneously and the sequential scheme that provisioned them one by one.

Yi et al. [254] proposed to solve the NFV multicast in multiple stages which included the traffic forwarding topology construction, function delivery topology construction and traffic steering. In particular, the minimum spanning tree was used to construct the forwarding topology while the function deployment was fulfilled by the backtracking strategy. After the first two stages, the traffic was steered through the deployed functions first before reaching destination. In particular, the main idea of it was the decoupling of traffic forwarding from function delivery, which allowed implementing them flexibly and independently in terms of minimizing the total cost. However, such decoupling also resulted in a new issue, that is, the paths between the forwarding topology and the deployed functions had to be determined, which would lead to extra overhead.

Considering the high complementation of SDN and NFV, many literatures tried to achieve a flexible and cost-saving multicast service provision in the integration environment of SDN and NFV. One typical example was Zhang et al. [248] which converted the distributed routing and function deployment problem into a centralized one with the global view provided by SDN. To solve such centralized problem, it presented an approximation algorithm (with an approximation ratio of 2), an exact algorithm based on branch-and-bound, and a dynamic heuristic method. For the approximation algorithm, it first searched for a single NFV node which was used by all the destination users and then it constructed a minimum spanning tree among the end users. The traffic was steered through the selected NFV node before reaching destinations using the constructed tree. The branch-and-bound algorithm was used to solve its proposed ILP model. Importantly, Zhang et al. [248] pruned the search space of NFV nodes and links, such that the total searching space for branch-and-bound was greatly reduced. The dynamic heuristic method was in fact based on the shortest path algorithm to connect the new joining users, which resulted in network congestion easily.

It is noticed that the bandwidth requirements may change after being processed by some VNFs. For example, the firewall may filter part of the traffic, such that the required amount of bandwidth may be reduced. In this regard, Zhang et al. [255] further divided the VNF-MC problem into three different cases, that is, the bandwidth requirement of any multicast traffic was *i*)the same; *ii*)increased; *iii*)decreased, after it was processed by NFV components. In particular, the three cases were solved by Zhang et al. [255] using approximation algorithms respectively. Nevertheless, the three approximation algorithms shared the same idea, that is, constructing the minimum spanning tree among all destinations and then creating a service chain to connect source to one destination using the shortest path. The difference among them was the weight metrics that they used as the analogue for bandwidth usage. Apart from constructing the multicast tree for a single multicast session, Zhang et al. [255] proposed a scheduling algorithm for handling multiple multicast sessions simultaneously. Since directly applying the proposed approximation algorithm for scheduling would cause imbalance in resource utilization, it proposed to reroute the traffic of overloaded links to those underutilized for the purpose of achieving certain extent of load balancing.

Moreover, based on the above researches, Zhang et al. [185] further studied the reliability of VNF-MC. Unlike Zhang et al. [248] and [255] that only constructed one service chain for the whole multicast group, Zhang et al. [185] proposed to separate all the destinations into sub-groups. For each sub-group, an independent service chain was constructed using the same strategy adopted by Zhang et al. [248] and [255]. Essentially, the sub-group suffered the same limitations as the whole multicast group did. Since any two sub-groups belonging to the multicast group did not share any deployed VNF instances, the cost for deploying more VNFs was inevitable in spite of the achievement of a certain extent of reliability.

6. Ongoing researches and challenges

NFV has been evolved from the proof-of-concept to practice. During the evolvement, many experiences and lessons are accumulated which can be used to avoid pitfalls as many as possible.

Despite this, there are still many challenges to be addressed before the full grown-up of NFV. Therefore, in this section, we jointly discuss the challenges faced and experiences learned on the road to NFV, and present them from bottom up.

6.1. Hardware design

Currently, there appear a lot of software based technologies that intend to accelerate network innovation from the application layer (i.e., software) instead of the infrastructure layer (i.e., hardware). NFV [24] and SDN [256] are two commonly known representatives among these technologies and they fulfill this target by decoupling software from the dedicated hardware which is replaced by the COTS based hardware. However, most of the already existing data plane functions are based on non-x86 architectures which are in the form of either merchant silicon packet processing or the expensive and customized integrated circuits [257]. The proprietary hardware is closely coupled with network functionalities, which results in network ossification and generates a significant challenge for moving to NFV. In order to solve this challenge, one simple and straightforward method is to replace the dedicated hardware with COTS based hardware once for all [258]. However, this operation not only causes great waste on dedicated hardware, but also leads to tremendous costs such as OPEX and CAPEX. Hence, it is not recommended and unrealistic. Current experiences focus more on collaborating such two kinds of hardware instead of using one of them solely (e.g., [259]). In other words, the proprietary network functions and VNFs can cooperate with each other to construct new services. Besides, we can also regard the proprietary hardware as the backup server, and use the COTS hardware to provide specific service functions. Nevertheless, the proprietary hardware will be gradually replaced by the COTS hardware on the way to NFV.

Despite the desire to migrate to a virtualized environment that is composed of COTS based hardware, the performance requirements of applications should be guaranteed. Specifically, the COTS hardware can support the requirements of many standard applications, and adopting COTS hardware in large scale can reduce network cost (e.g., CAPEX and OPEX) greatly. However, the COTS hardware is relatively weak in terms of offering high performance on throughput and reliability [260], etc. Considering the dynamic requirements of enterprises and operators, the hardware should be designed with all the potential situations covered. In addition, directly migrating existing network functions and applications to general-purpose servers without considering their features and demands may lead to unpredictable results [261].

In order to address the issues suffered by COTS based hardware, two major methods are used. The first one is to use data plane acceleration technologies (e.g., DPDK [115] and SR-IOV [46]) and the second one is to use high performance hardware (e.g., IBM RackSwitch [92] and Cisco Nexus Switch [144]). With respect to the acceleration technologies, they are already elaborated in the previous section and are usually applied to the general-purpose hardware (e.g., x86 server) for providing high performance and predictable operations. In particular, these technologies and other VNFs are installed on x86 servers as software module which is similar to the mechanism of cloud model [186]. However, we should be aware that NFV and cloud are actually two different entities, that is, NFV focuses on function virtualization, while cloud focuses on resource virtualization. In this way, the design of COTS hardware should be different from that of cloud model. With respect to the high performance hardware, it is usually purpose-built [25]. Although the purpose-built hardware may not be able to provide as high flexibility as the COTS hardware does, it can satisfy the rigorous demands of applications and services. For example, in order to achieve significantly high performance, the Application Specific Integrated Circuits (ASICs) are usually designed using purpose-built hardware [262]. Despite this, another drawback of purpose-built hardware is high cost compared to the COTS hardware. As a result, the high performance offered by purpose-built hardware is actually obtained by sacrificing some overhead and flexibility. Hence, based on the actual requirements of customers, the key point of designing suitable hardware for NFV is how to determine the trade-off among performance, cost and flexibility, etc.

Many layer 4-7 network functions such as load balancing and DNS can work well on the COTS based hardware, because their requirements on packet processing and interface speed are not high [263]. However, for those functions (such as data center switching and gateways) with high I/O speed and performance requirements, the COTS hardware is not a good option. Instead, they still rely on specialized hardware which can provide higher I/O performance than the COTS based hardware does [264]. Besides, given one kind of server, it may be offered by many companies or vendors. In this regard, how to determine the most suitable one is another issue. Currently, the factors used for hardware selection usually include cost, service quality, latency, preferences, reliability, scalability, security, etc. [265]. Considering these reasons, most enterprises and operators would like to start out with COTS based hardware first to build their NFVI environment, and then gradually adjust their workload to high performance proprietary hardware in order to satisfy some high performance requirements.

6.2. VNF deployment

VNF plays an important role in the whole NFV architecture as NFV initiators intend to generalize the underlying proprietary hardware and implement the corresponding network functionalities in the form of software (i.e., VNF). Importantly, with more efforts from the equipment vendors and service providers contributed to the development of VNF, there evolve many critical challenges and the primary of them is determining the required number of VNFs and the locations to place them. Besides, it is hard to find the optimal solutions especially in the large scale network scenarios due to the NP-hard characteristics of the VNF placement [191]. As explained, the solutions for VNF placement can be categorized into exact and heuristic ones. The former offers optimal results and extremely high running time [23], while the latter offers suboptimal results and low running time [187]. Therefore, how to make the trade-off between them is still an open issue.

Generally, given a small network (fewer than 1k hosts), it may contain roughly 10 proprietary functions while in a very large network (more than 100k hosts), the number of proprietary functions may reach about 2000 [2]. Thus, the proprietary network functions are widespread in the network. Under this situation, the coexistence between VNF and proprietary network function is inevitable and critical for NFV service provision. Accordingly, the solutions for NFV service provision should also take them into consideration, since there is no need to place a new VNF where there is already one proprietary function playing the same role. Although some literatures have verified their algorithms over the VNF and proprietary function hybrid environment, they did not specify how to cooperate the two kinds of network functions [184].

Despite the same underlying COTS based infrastructure, different VNFs may have different deployment requirements on instantiation time, cost, etc. For example, the Minios of Xen could be used to enable a fast instantiation (around 31 milliseconds) for VNFs [266]. However, in order to support the large scale VNF deployment, the VNF deployment should be not only fast, but also automated and intelligent. In this regard, many communities have already started researching the potential of applying the technology of machine learning into SDN and NFV due to its healing and automation features [267]. In particular, the large amount of data from users can be collected and analyzed by machine learning en-

gine for making decisions and facilitating the VNF deployment. This ultimately leads to an intelligent network where services are spawned automatically by launching VNFs from the repository [267]. Nevertheless, machine learning requires a long time of learning before it is effective enough, and how to live through this period should be thought carefully.

In addition, the VNF validated for functionality and performance on certain NFVI cannot guarantee that it would function properly on another NFVI [268], because NFVIs differ in several parameters (e.g., CPU core allocation policy and storage) which may affect the performance, availability or even the proper functionalities of some VNFs. Certainly, we can deploy VNFs into NFVI with the requirements not fully satisfied. In doing so, the results are unsatisfying, which may decrease some performance in the best case or crash unexpectedly in the worst case, and the crash situation is very hard to debug [269]. The NFVI parameters can be configured to satisfy the requirement of VNFs. However, such configuration or reconfiguration process may need some manual work and a long processing. Targeting on this, many solutions are proposed to automate the parameter configuration process. One of them is OpenStack API [270]) which translates the VNF requirements into API calls to create necessary virtual resources for VNF deployment in NFVI. Nevertheless, given a NFVI which was configured correctly for one VNF, when another VNF with different requirement is deployed on this NFVI, the administrator of this NFVI may have to make configuration changes, which may no longer provide an optimal environment for the original VNF [271].

Another phenomenon slowing down the deployment of NFV is the contradiction between VNF vendors and service providers [230]. In order to support the fault-finding requirements and enable smooth deployment of NFV, the service providers would like to have the capability of packet level traffic visibility for diagnosing unexpectedly occurred outages, while such capability is not offered by equipment vendors. One emerging approach to solve this problem is the Tap as a Service (TaaS) [272] which offers the remote port mirroring capability for accessing data inside the virtualized network environment without involving the vendors. However, TaaS requires copying packets entering into or leaving from such environment, which can result in large amount of traffic and lead to network congestion.

In fact, the VNF deployment can be regarded as a kind of dynamic optimization problem. Most of the existing algorithms are not general enough to fit all the common network scenarios. Besides, to fulfill the specific service requests, node resource (e.g., CPU, storage and memory) and link resource (e.g., bandwidth) consumptions are inevitable but can be reduced significantly by using efficient strategies. For instance, the node resource consumption can be reduced by serving the given requests with the minimum number of VNFs. However, such decision may result in long paths from the source node to the destination. In this way, the network bandwidth resource consumption increases. On the other hand, the bandwidth consumption can be decreased by placing the required VNF instances near the source node or the destination node such that the traffic can follow the shortest path to destination rather than the detoured path. Obviously, such behavior would lead to high resource consumption on nodes. Hence, the trade-off among these resource consumptions should be carefully designed on the basis of the actual application situations.

6.3. VNF life cycle control

NFV has introduced a variety of new entities to the telecommunication networks by decoupling the software implementation of network functions from the proprietary hardware [24]. All the entities and their relationships are uniquely defined within the NFV system. Among them, one significant entity generated by such decoupling is VNF which can increase the flexibility of service construction and reduce the network cost such as CAPEX and OPEX [273]. However, to fulfill these benefits, the life cycle management of VNFs must be addressed properly, since it involves many aspects, for example, the virtual resource allocation and recycle.

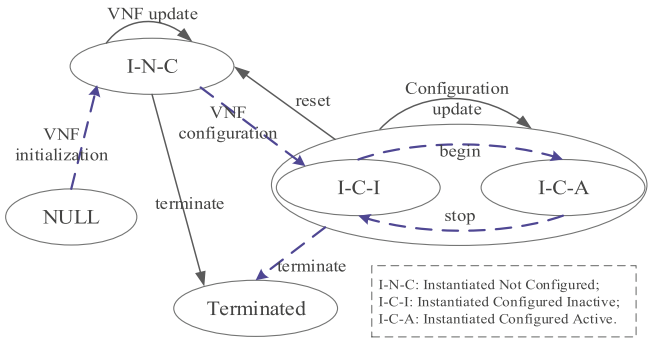

Fig. 13. The VNF instance status transition diagram [54].

There exist five transformation status for any VNF during its life cycle according to the VNF reference architecture given by ETSI [54], and they are NULL, Instantiated Not Configured (I-N-C), Instantiated Configured Inactive (I-C-I), Instantiated Configured Active (I-C-A) and Terminated respectively. Among them, the status of NULL and Terminated indicate the beginning and end of one VNF instance respectively. Then, the I-N-C status means that the VNF is prepared with all required resources allocated, but waits for proper configuration before going to the next status. The I-C-I status indicates that this VNF is well configured and ready for service provision, while the I-C-A status means that this VNF is working in process. Besides, to provide an integrated view, ETSI [54] presented a transition graph among these five VNF instance status. We redraw the graph and show it in Fig. 13, in which the blue dotted line indicates a complete process of a VNF from instantiation to termination.

However, such abstract design is not enough for real implementation. Thus, the enterprises and operators develop their own description of VNF life cycle management. One typical example is an elastic service controller proposed by Cisco [274], which explains the life cycle management of VNF in the sequence of onboard, deploy, monitor, scale, heal, and update. Comparatively, Cisco adds several new aspects, that is, i)onboard, which enables to define any new type of VNFs, ii)monitor, which tracks VNF performance metrics such as CPU usage and memory consumption, iii)heal, which provides fault recovery for VNFs. Meanwhile, Juniper also provides a VNF life cycle management tool which considers more aspects during the life cycle management process, e.g., resource planning and VNF image management [275].

The entities of NFVO and VNFM within NFV are in charge of managing the life cycle (e.g., instantiation, update, query, scaling and termination) of VNFs. Currently, the boundary between NFVO and VNFM is blurring, such that the management task is usually carried out by the two of them. In particular, the functionalities of NFVO and VNFM have already been implemented by some existing tools such as Tacker [127] and Cloudify [128]. These state-of-the-art NFV orchestrators have already been discussed in Section 4.2. Despite the fact that many orchestrators have explored the VNF management functions extensively, there does not exist a unified interface or framework to cooperate these orchestrators [26]. Considering the cross-domain and multi-vendor characteristics of VNFs [276], the heterogeneity among various NFV orchestrators should be carefully addressed. Besides, in order to well perform the life cycle management, these NFV orchestrators have to exchange the VNF related information with the element managers that directly

connect to VNFs, thus to obtain enough information for a complete description of VNFs. In this regard, Network Configuration Protocol (NETCONF) [277] and libvirt [105] offer different ways for VNF description based on XML descriptor file and extensive virtualization features respectively.

Typically, VNFs are orchestrated in a virtualized environment provided by VM [161] or container [103]. However, in some specific circumstances, VNFs are deployed on physical servers directly and managed by hardware based monitoring management system [135], [278]. However, such design makes the VNF life cycle management much more difficult than managing VNFs in the virtualized environment, because the hardware environment may not provide as flexible management as the virtualized environment for VNFs. Nevertheless, the service consistency must be guaranteed no matter in physical or virtual environments.

6.4. Service chaining

The service chaining problem is NP-hard, since it is evolved from the VNF placement and VNE problems which have already been proved to be NP-hard [191]. This problem includes not only dynamically placing the required functions (i.e., VNFs) at suitable locations, but also steering traffic through these placed functions. Thus, it suffers from many challenges. Most of current service models (e.g., [279] and [280]) rely on using the network functions that are coupled with network topology and physical resources to provide services, which is static and rigid. On one hand, such static nature limits the ability of introducing new or modifying existing services and network functions. On the other hand, the rigid situation will result in a cascading effect, that is, the change of one or more functions in a service chain will affect other functions in this service chain.

As explained in Section 5.4, there are a lot of deterministic and heuristic methods proposed to solve the service chaining problem. However, considering the fact that the network is becoming larger and larger, researchers gradually turn their attentions to heuristic methods, because using deterministic methods to solve the large or even super-large scale network service chaining problem in reasonable time is almost impossible [187]. In contrast, the heuristic methods are much more time-efficient. For example, Pham et al. [281] proposed a heuristic service chaining approach which can be solved within polynomial time. Despite the great difference between the execution times of heuristics and exact solutions, we should be aware that the execution times of different heuristic methods do not differ too much. In this regard, the heuristic solutions focus more on other performance metrics such as energy consumption and reliability [231], etc.

Reviewing the service chaining work introduced in Section 5.4, most of them followed specific greedy mechanisms which may result in the situation of local optimum. In addition, adopting greedy strategy may lead to more resource fragments such that it may be hard to satisfy the following requests with high resource requirements. Thus, apart from leveraging the greedy strategy, other strategies should be developed. Moens, and Turck [195] proposed a customized model for managing the variability of SFCs in the hybrid environment where VNFs and dedicated functions coexisted. Besides, it also demonstrated the reduction in service deployment cost and the increase in resource utilization. However, the proposed approach was only studied and evaluated in a small network. Eramo et al. [210] proposed a consolidation approach based on VNF migration and back-to-back mechanisms for SFC deployment. Scheid et al. [282] proposed a policy-based approach for automated SFC construction with minimum disruption. The two of them could be applied to both the homogeneous (VNFs only) and the heterogeneous (VNFs and physical middle-boxes coexistence) environments. Mechtri et al. [283] even proposed an eigendecomposition-based method for the placement of both virtual and physical network functions in cloud environments.

The scalability is another critical issue that should be solved for service chaining. Specifically, VNFs should be able to be added to or removed from the existing SFCs dynamically. Despite the fact that several of the above mentioned work (e.g., [134] and [283]) demonstrated the scalability for their models and schemes, they actually meant that their models and schemes could be applied to networks with different sizes and heterogeneities [284]. Besides, the service chain is composed of multiple VNF instances which may come from different vendors [285]. Typically, different vendors have different standards and criteria. Hence, how to coordinate them to form and provide a high qualified service is challenging and urgent to be resolved.

6.5. Performance evaluation

The conventional network is composed of dedicated equipment, such that the performance evaluation is usually fulfilled by using a dedicated environment or facility. Instead, NFV virtualizes network functions into software and runs them on COTS hardware. By doing this, NFV can be investigated and evaluated in a generic manner since COTS hardware brings many benefits such as high flexibility and scalability, low cost, etc. However, these benefits are obtained at specific expenses. For example, i)bottlenecks of the data plane may appear due to the virtualization of network functions; ii)the resources allocated to VNFs may be over-provisioned since the VNF performance is hard to be predicted.

Packet forwarding process is critical to deal with the workload of many telecom network subsystems such as EPC [287] and RAN [174]. According to Gallenmller et al. [288] and Luis et al. [289], there are a lot of factors affecting packet forwarding, such as bandwidth of Ethernet NICs, CPU speed, peripheral component interconnect express and memory bandwidth. Most existing works are under the assumption that the VNF performance is bounded by the CPU speed and the Ethernet bandwidth. This is true for NFV as well. For instance, when deploying NFV, hundreds or even thousands of the related functions have to be virtualized on one single server or across multiple servers [290]. Such behavior requires a large amount of Ethernet bandwidth to ensure the performance when virtualizing network functions. For example, with respect to the short packet size (64B), the minimum network I/O capacity required is 14.4MPPS for achieving 10Gbps throughput on a COTS server with a general NIC [291].

NFV enables to accelerate the packet forwarding performance by virtualizing the NIC over hypervisor based networks. In addition, by applying data plane acceleration technologies (e.g., DPDK and SR-IOV) to the virtual NIC, the performance of virtual NIC could be improved further. Considering this, Nakajima et al. [291] proposed a high-performance virtual NIC framework for NFV, in which the DPDK-compatible APIs were provided. It demonstrated that the virtual NIC could achieve over 120 Gbps throughput and over 14.2 MPPS I/O processing. Kourtis et al. [46] claimed to achieve 81% higher throughput than the native Linux kernel, by applying DPDK and SR-IOV on virtual NIC. Despite the high throughput achieved by using virtual NIC, it also resulted in high bandwidth and energy consumption. A large amount of bandwidth was required in order to guarantee the performance of virtualizing hundreds or even thousands related functions [292]. Such contradiction would inevitably cause challenges for NFV. On one hand, NFV indeed improves the performance, on the other hand, more attention should be focused on the trade-off between performance and overhead as different enterprises have different objectives.

For most of current telecom operators, their network architectures are still fairly static. However, the introduction of VNF can relieve such situation. For example, the network load balance can

Table 5
Processing time estimation of network functions (second) [286].

Network functions	Proprietary hardware based	COTS hardware based
Load balance	0.2158	0.6475
Firewall	2.3590	7.0771
Virtual private network	0.5462	1.6385

be achieved by the migration of some VNFs. In this way, the VNF performance evaluation is very important. Most existing works announced their benchmarks from different perspectives to evaluate the performance of VNFs. For example, IETF described general metrics, strategies and benchmarks for evaluating and testing VNFs [293], while other individual researches focused on evaluating specific VNFs. For example, Lange et al. [294] studied the VNF benchmarks in terms of the use case of LTE SGW. However, it is still challenging for VNFs running on COTS hardware to offer comparable or even better performance than the functions running on proprietary hardware. For example, given the amount of packets, proprietary network functions generally require less processing time than the VNFs. Table 5 presents the data of packet processing time with respect to three network functions (load balance, firewall and virtual private network functions) and it is easy to discover that the difference between the processing time of proprietary hardware based firewall and the VNF-based firewall is the biggest (roughly 6 s) and that of the load balance is the smallest (roughly 0.4 s) [286]. Besides, mechanisms must be developed in order to make sure that these VNFs are portable among servers. Fortunately, ETSI has tested several NFV use cases (e.g., DPI) in a fully virtualized environment and achieved high performance on throughput (around 80Gbps per server) [295].

Another fact which should not be ignored is that the VNFs may have different behaviors based on the underlying hardware, operation systems and implementation solutions, which results in the unpredictability of VNF performance. For example, a VNF may report the network connectivity problem with another one where in fact the real problem is the lack of sufficient CPU cycles to handle the keep-alive messages. Some specific VNFs require careful configuration even though they are implemented to be hardware independent. In order to satisfy the real-time feature of voice and video related VNFs, it is better to adopt several time synchronization mechanisms or even disable some conflicting features [296]. Besides, the VNFs are generally not as reliable as the proprietary functions. Hence, the vendors should also offer redundancy for their VNFs in case of failure and the typical way to achieve redundancy is to configure multiple connections between the ports and network interfaces [297].

Traditionally, the usual way for guaranteeing the network performance is to estimate the peak demands of network functions and then allocate the corresponding amount of resources to them. However, this method may not work well for VNFs on the shared COTS-based hardware because the VNF performance is unpredictable and performance interference between different VNFs is hard to be avoided when they cooperate with each other. Hence, how to guarantee the VNF performance, especially in large scale carrier-grade network, challenges NFV greatly. As explained, using the data plane acceleration tools can improve the performance of VNFs, e.g., Lange et al. [294] and Kourtis et al. [46] investigated the possibility of using DPDK and SR-IOV to enhance the VNF performance respectively. Despite the achievement of high performance claimed by DPDK and SR-IOV, they also result in some energy consumption and security issues which are not covered in most works.

Many research works focused on building the performance prediction model for VNFs. Suksomboon et al. [298] developed a simple mathematical model for the VNF performance prediction on multi-core processing systems and Ref. [299] concentrated on using a simulation model based on NS3 [300] for prediction, whereas Xu et al. [301] and Dobrescu et al. [286] leveraged the cache miss for prediction. Although different, these methods shared the target of quantifying the performance of VNFs. Ref. [302] investigated on network performance prediction and it proposed an analytical approach to model the network with NFV capabilities. As demonstrated, it could forecast the network performance with given configurations in both general and specific network scenarios. However, it only considered one single traffic model, which indicated that its forecast might not be general. From the perspective of Service Level Agreement (SLA), Sun et al. [259] proposed the SLA-NFV framework which leveraged a hybrid infrastructure (consisting of software and programmable hardware) to enhance NFV's capability in terms of various SLAs. In spite of these efforts, most VNF performance prediction models suffer from the prediction accuracy issue. Besides, error predictions can easily mislead the operations, which causes the performance degradation or even crashes the system. Therefore, how to compensate for the potential performance loss and how to cope with the potential emergencies, are key requirements for guaranteeing the VNF performance.

Various NFV performance enhancement schemes and techniques have appeared. Whether they are applicable for the practical NFV situation is hard to say now and most of them may even cause performance degradation. Therefore, on one hand, with respect to network operators and service providers, they have to be responsible for choosing the most suited responsibilities for their infrastructure and service models. Some schemes may even be combined to offer a possible best solution. On the other hand, for the researchers, apart from keeping studying the way to improve NFV performance, other potential and necessary directions may include swapping mechanisms between performance and other factors (e.g., flexibility and scalability) as different enterprises have different focuses, and specifying the detailed process of the phased migration to NFV.

6.6. Policy enforcement

Policy enforcement is an old term which is now being used to describe a variety of technical solutions [303]. For example, from the perspective of network, we can regard policy enforcement as a kind of network access control application which uses a set of rules or parameters that are known as the 'policies' to restrict the users in legal regions [304]. From this point of view, policy enforcement can be used to solve some specific security issues. However, within NFV scope, policy enforcement can perform more than just security functions. For example, policies can be determined to enforce business rules and specify resource constraints [22] in the NFVI environment.

Typically, the network policies can be taken into account in the framework of general cloud services and used in individual subsystems or specific use cases [22]. For example, Ma et al. [305] determined the subsystem policies for achieving a load balanced network, Mayer et al. [306] proposed a network policy to support the information exchange and cross-domain management, and [307] focused on optimizing the distributed virtual network computing with policies in terms of VNF placement and migration. However, an open challenge is how to leverage the existing efforts of policy enforcement to satisfy the unique requirement of NFV [308], because on one hand, policy enforcement is an old term while NFV is relatively a new concept. Such contradiction would inevitably introduce incompatibility between them. On the other hand, NFV introduces a lot of new concepts such as VNF and MANO, which should be carefully considered when applying the mechanisms of policy enforcement.

In order to address these problems mentioned above, a lot of researches have been done, which cover many aspects such as policy definition and framework. Firstly, with respect to the policies defined in NFV, they are classified into global and local ones based on their working scopes [22]. For example, some well-known SDN controllers such as OpenDaylight [136] and Floodlight [309] can be regarded as local policy engines, since they can define and enforce network policies in their controlled domains. Most of the policies are enforced in a centralized manner especially with the appearance of SDN, because SDN provides an intelligent center for guiding the process of policy enforcement. With the centralized view of the underlying network, the controller can easily decide when, where and what policies to be enforced. For example, Ref. [310] proposed an orchestration framework based on SDN/NFV to enforce the network function policies and offer certain extent of service quality, while Ref. [311] presented a network management approach which exploited the SDN principles to decouple the policy resolution from the policy enforcement.

Due to the fact that centralized control may lead to the single point failure, the distributed policy enforcement is explored by many researches. For example, Wang and Minsky [312] studied the policy enforcement in social network in a decentralized way, while Ref. [313] fulfilled this in the satellite network. In particular, the former proposed to establish the global network policies only, while the latter built both the global and local policies. In this regard, Araguz [313] could achieve better performance than Wang and Minsky [312] with the price of complexity. In addition, the distributed policy enforcement generally involves network policy space analysis, where the set operations consume most of the computation. Fortunately, many works have already been proposed to solve this problem. For instance, Li et al. [314] proposed an algorithm based on atomic hyper-rectangle indexing for fast policy space set operations.

Currently, most of the policy enforcement frameworks proposed to serve NFV are actually based on the NFV integrated open source projects (typically OpenStack). One example can be found in [315] which used an OpenStack module as a policy engine to support the policy monitoring and energy management, while Ayache et al. [316] focused on implementing the access control policy enforcement in the cloud environment by using OpenStack. However, these works focused on the process of policy enforcement and let OpenStack finish all the other related processes such as policy format definition and verification. This naturally led to a strong coupling between them, which might reduce the flexibility. Many customized policy frameworks were also proposed to achieve fine-grained policy enforcement. Based on the reference architecture of SDN and NFV, Lorenz et al. [317] proposed an integrated solution, in which a fine-grained security policy enforcement was provided. In particular, Lorenz et al. [317] examined the stateful firewall with different integration approaches. Comparatively, Shaghaghi et al. [318] intended to obtain a policy-enforcement-as-a-service model which had a ‘defense in depth’ protection and could drop unsuccessful access requests before engaging the data provider.

Generally, there are two situations of policy enforcement that should be known. Firstly, network policies are usually implemented and enforced with high priority given to the traffic sensitive to network conditions [307] such as bandwidth, delay, etc. For example, for the time sensitive traffic, the policies for it may not be effective if the policy enforcement is executed after a long time. Secondly, one kind of policy is usually enforced in terms of one kind of scenario [285]. For example, Fayazbakhsh et al. [319] developed an extended SDN architecture and defined many policies specifically used as tags of packets, which could provide necessary and causal information for achieving fast service provision and low cost. Similarly, He et al. [320] used policies as tags to distinguish different instances that traversed by the same packet at the same ingress port on the same switch, thus to avoid the service path loop that may be generated by using specific routing algorithms. Therefore, the difference between Fayazbakhsh et al. [319] and He et al. [320] was that the former tagged the policies on packets while the latter tagged the policies on middle-boxes.

Deep in the NFV framework, the policies are stored in local databases so that they can be accessed by the local policy engine [22]. Apart from the already existing policies, other new kinds of network policies are putting into the databases gradually. However, the old policies may have conflict with the new ones, which leads to another serious challenge [285]. Such conflicts happen because different policies with the same priority-levels are enforced to the same entity, and hence it is hard to decide which one should be adopted. For example, a new policy declares that “customer A does not have the ability to use the specific kind of VNF (e.g., DPI)”, while one existing policy allows customer A to use the VNF DPI. Directly using one policy to cover the other may generate unexpected errors which are hard to debug [321]. Therefore, the policy conflict detection mechanisms and the corresponding solutions are necessary and shall be focused with more attention. Furthermore, policy exchange would also result in challenges for NFV. Because different policies in different administrative domains have different configurations and parameters, and inappropriate exchange action would violate the normal functionalities of them.

6.7. Energy efficiency

Energy consumption has always been a cumbersome problem in communication networks such as data-center network, Cloud RAN (C-RAN) and EPC. The main consumed energies are electricity and fuel. Among them, the proportion of electricity can generally reach about 85%. For example, the total electricity consumption of China Telecom reached 65 billion kilowatt-hour in the year of 2011, which consisted of communication (50%), cooling (40%) and lighting (10%) [322]. In addition, the number of middle-boxes in the network is comparable to the L2/L3 forwarding devices and all these devices account for about 15% of the total energy consumption [40]. NFV demonstrates its potential on reducing energy consumption by consolidating equipment and exploiting the power management feature in standard computing and storage servers. For instance, we can switch off or put some of the servers into an energy saving mode and consolidate the workload on a small number of servers during the off-peak hours (e.g., middle night) by using virtualization technologies. According to the study of ETSI, NFV can potentially deliver up to 50% energy savings compared with traditional appliance based network infrastructures [57]. However, this statement has not been explicitly verified yet. Besides, the cloud-based NFV [20] has attracted people’s attention nowadays. The energy consumption in cloud is extremely high and is expected to increase by 63% by the year of 2020 [323]. On the way to the deployment of NFV, the demands for network devices (e.g., servers and switches), power-supply systems and base stations will definitely increase sharply. In order to satisfy the customers’ demands at any time, the 24-h services are usually required. All the situations mentioned would lead to tremendous energy consumption. Therefore, it is worried that using cloud technologies to fulfill NFV would aggravate energy consumption.

To solve the energy problem in NFV, GWATT [324], a tool proposed by Bell Labs, can be used to decrease the energy consumption by virtualizing network functions. Mijumbi et al. [325] used the tool to estimate energy savings expected by NFV in three main use cases: virtual EPC, virtual RAN and virtual Customer Premises Equipment (CPE) respectively. The virtual EPC could have 24044.1 MWATTS power savings, virtual CPE had 2703.63 MWATTS and virtual RAN had 26604.4 MWATTS savings on the basis of default GWATT settings [24]. Instead of estimating energy savings in use

cases, Xu et al. [326] conducted a measurement on power efficiency of three different NFV implementations which are DPDK-OVS, Click Modular Router and Netmap. In addition, based on the measured results, more power-efficient NFV implementations in terms of software data plane, virtual I/O and middle-boxes can be built. Likewise, Tang et al. [327] also used the estimated results to guide the design of power saving strategies. The difference was that Tang et al. [327] conducted the energy consumption in a disaggregation manner instead of aggregation. Furthermore, Krishnan et al. [315] proposed an open stack based solution for the energy management in the NFV-based data centers, and it claimed to optimize the energy consumption by using periodic monitoring and dynamic resource management mechanisms.

The integration between NFV and other paradigms (e.g., SDN) is another effective way to reduce energy consumption and improve energy efficiency. Taking SDN as an example, the centralized network view and control offered by SDN can be used to effectively manage and monitor the NFV networks, which in turn save energy consumption. Bolla et al. [328] extended the open source framework Distributed Router Open Platform (DROP) and presented a higher version DROPv2 which enabled a novel and distributed paradigm for NFV via the integration with SDN. In order to meet the increased demand on energy efficiency, DROPv2 introduces some complicated power management mechanisms by means of Green Abstraction Layer (GAL). From the perspective of architecture, DROPv2 provides the network data plane and control plane capabilities by consolidating a large number of well-known software projects, rather than operating as an independent entity. Besides, Luo et al. [329] proposed an energy efficiency scheme using SDN and NFV, which included monitoring infrastructures, controlling network topologies, and saving energies.

Considering the wide recognition and benefits brought by SDN and NFV, we recommend more efforts should be contributed on adopting the integration model of them to handle the energy consumption problems. Nevertheless, the energy consumption is also an issue for SDN especially in the large scale network scenarios. In addition, the energy efficiency improvement may lead to network performance degradation, and the trade-off between them should be carefully considered.

6.8. Reliability

Reliability represents the capability of a system against hostile or unexpected situations and it should not be affected when transforming from the traditional network to the NFV based network. Within the scope of NFV, network functions are virtualized as VNFs. The virtualization actually introduces many unique challenges to NFV [62], which makes the reliability much harder to achieved.

Although the proprietary functions may be failed due to many reasons such as mis-configuration and overload, many traditional network operators and equipment vendors can still guarantee high reliability (above five nines or 99.999%) by using these proprietary functions to provide services and making sure that the failure detection time is less than 1 s. Therefore, in order to guarantee the service reliability in NFV, one obvious challenge is to ensure that the VNFs are more reliable than or at least the same as the proprietary hardware based functions [330]. On one hand, the NFV components (e.g., VNFs) should offer better performance on many aspects. For example, the success detection rate for failures should be improved and the failure restoration time should be reduced. On the other hand, the design of NFV components (e.g., VNFs) should take the COTS hardware and the virtualization into consideration, because they are highly associated for accommodating these components. The usual methods for NFV to keep reliability are to improve the resilience and elasticity of VNFs. However, such behavior may lead to other new failure points which require to be conducted automatically [25].

As VNFs are used to construct services in NFV, the research area is gradually shifting from the single VNF reliability to the end-to-end service reliability. According to the standard of carrier-grade VNFs, equipment vendors and service providers would like to fulfill the service reliability with three kinds of Service Availability Levels (SALs). Specifically, the default setting for the highest SAL should be less than 1s' failure detection time (equal to that of proprietary functions) and 5-6s' recovery time, the middle SAL includes less than 5s' failure detection time and 10–15s' recovery time, and the lowest SAL is less than 10s' failure detection time and 20–25s' recovery time [62]. Nevertheless, in order to deliver the VNF-constitute services with reliability guaranteed, many other aspects should be carefully addressed, which include how to avoid the single point failure (including failure detection and prevention) and how to achieve a fast recovery from failure, especially in the multi-vendor environment since VNFs may be supplied by different vendors [331].

The end-to-end service continuity is another important aspect for service reliability. To guarantee the service continuity, VNFs must be able to preserve the related state information which can be used to protect customers from disruptive events and to recover services from disasters quickly [332]. In addition, to make sure that services are still available when adopting NFV as the architecture, network operators need to adjust and configure various parameters on their servers according to practical situations and criteria. Although network operators cannot guarantee all the services to function well when a large scale network disaster happens, they can at least guarantee some essential services. For example, the voice call service, which enables emergency events to be received in time rather than the online game service, can be guaranteed by transferring the resources that originally allocated to the online game service to the voice call service. Furthermore, redundant and remote VNF deployment is also necessary for cases of emergency and disasters.

6.9. Security

Security is always an important focus no matter in conventional network scenarios or NFV supported network scenarios. Since the virtualization technologies used in cloud computing are also applied in NFV to provide a virtualized environment, the challenges caused by these virtualization technologies are also faced by NFV. ETSI has established a Security Expert Group (SEG) to particularly concentrate on identifying and resolving the security issues in NFV. Besides, the potential scale of security problems that NFV might introduce is another aspect investigated by ETSI SEG. After the assessment of SEG, the results show that NFV indeed introduces some new security problems such as topology validation and multi-administrator isolation. The potential security threats investigated by ETSI SEG are listed in Table 6. However, none of them are intractable [333]. For example, with respect to topology validation, Jaeger [334] proposed a SDN based security orchestrator which enhanced the ETSI NFV reference architecture with a widespread trust management and offered a global view for fast and efficient topology validation.

NFVI is the foundation for the deployment of NFV. However, it suffers from both internal and external threats [336]. On one hand, the internal threats result from inappropriate operations of users, which can be avoided by following strict operation procedures. On the other hand, the external threats are from vulnerable design and implementations, which are difficult to be avoided. Considering that most security issues result from decoupling functions from the proprietary equipment, the hypervisor is the one which introduces many new threats, because hypervisor can be viewed

Table 6
The summary of potential new security concerns for NFV assessed by SEG [63].

Security Threats	Description
Topology Validation & Enforcement	Checking if the topology design satisfies the desired security goals.
Availability of Management Support Infrastructure	Remote management on any large computing or network infrastructure deployment.
Secured Boot	Validation and assurance of boot integrity.
Secure Crash	Security concerns caused by unknown and unexpected state.
Performance Isolation	Isolating common resources like CPUs, memory and storage.
User/Tenant Authentication, Authorization and Accounting	Using identity and accounting facilities in the network.
Authenticated Time Service	Identifying the integrity of timing messages.
Private Keys within Cloned Images	The key pair used by images running on VMs.
Back-Doors via Virtualized Test & Monitoring Functions	Illustration of the risk on NFV debug and test interfaces and providing protections where these interfaces are required.
Multi-Administrator Isolation	Distributing different privileges to different administrator hierarchies.

as the potential security surface on which VNFs may be attacked in the form of compromised isolation. Meanwhile, the solutions for hypervisors (and other NFV components) are provided by multi-vendors. The potential incompatibility and confliction among them also increase the security risks [336]. Moreover, as ETSI aims at building an open and diverse ecosystem for NFV, the components of NFV architecture are likely to be provided by different equipment vendors or service providers. Such hybrid situation may result in security loop holes when integrating different components or solutions [15], due to the lack of standard inter-operability specifications.

To solve these security issues, most researchers and enterprises are relying on integrating SDN and NFV to provide a comprehensive security framework. For example, based on SDN and NFV, Liyanage et al. [337] improved the security performance by 50% compared with traditional methods, while Park et al. [338] achieved a fast recovery (less than 2 s) from interrupted security services. With regard to the Distributed Denial of Service (DDoS, about 65% higher than the other threats), SDN and NFV provide efficient and scalable features and mechanisms for detecting malicious activities and preventing them from spreading or disrupting [339]. Leveraging the continuous monitoring and centralized management offered by SDN, many security threats can be detected quickly. However, the SDN controller can easily become a single failure point recognized by almost half of the investigated security experts [335]. Therefore, how to mitigate the drawbacks and leverage the benefits introduced by SDN is critical for guaranteeing the security of NFV. In this regard, the technology of blockchain [340], a distributed technology used to secure the digital currency, is recommended to compensate for the drawbacks resulting from the SDN centralization. Due to the success of applying blockchain into the bitcoin networks, the same situation may also be possible for SDN and NFV based networks.

In addition, a lot of tools and practices are employed to solve the challenges brought by the integration of SDN and NFV. According to the security report of ARBOR [341], the representative one is Netflow (a kind of network monitor) based analyzer which has the highest usage proportion (up to 78%), followed by firewall related tools (64%) and IDS/IPS related tools (51%), etc. Unfortunately, these tools are usually virtualized across racks and data centers, which results in blurry boundaries among different security domains. Thus, the automation mechanisms for placing these corresponding functions are required in order to manage and adjust the security domains in time. Huawei highlighted the importance of an effective security monitor for discovering threats and mitigating attacks in terms of NFV security [342]. Wedge Networks described the existing security threats in NFV and presented the corresponding mitigation methods [343]. In fact, the work of Wedge Networks is also done by many other enterprises such as Alcatel-Lucent and Intel via employing a lot of useful tools and practices [344]. Although these works can address the security issues to a certain extent, they are complex for real implementation.

Despite the large amount of security threats in NFV, there are three common aspects that service providers typically adopt to address the existing NFV security challenges and achieve higher security level than before. The first one is the virtualization of PNFs, which enables network operators the ability to build a smart and flexible network to prevent threats like denial-of-service attack. The second one is the automation capability offered by SDN and NFV, which enables service providers the ability to manage and configure security network functions dynamically. The last one is the consistent monitoring and analysis ability, which can help protect users against some malicious activities before they cause the wide damage. For example, we can prevent users from being eavesdropped by attackers via comprehensive and continuous monitoring.

7. Future directions and application scenarios

Most network owners and operators have been relying on a handful of vendors that provide proprietary and vertically integrated hardware, operating systems and control features with limited or no programmability. The lack of network programmability, control inflexibility, and the need for network operators to adapt tedious and error-prone manual configuration methods to provision and manage network services has led to increased operational complexity and prolonged time to market for new services [345]. Fortunately, NFV and SDN bring opportunities to solve such challenges by leading a trend to network softwarization [346] which is expected to revolutionize the way network and computing infrastructures are designed and operated.

Due to the benefits (e.g., network innovation and service diversity) enabled by network softwarization, many network owners and operators are already on the road to a softwarization network, which can be reflected in three aspects, that is, *i*)a majority of networks expand based on the COTS hardware; *ii*)new services are constructed by deploying new software elements instead of proprietary hardware based ones; *iii*)the automation of service deployment and provision is popular, which can reduce the time to market and the OPEX. Therefore, it is necessary and meaningful to discuss the future directions and application scenarios of NFV from the perspective of network softwarization.

7.1. Network softwarization

Generally, network softwarization indicates an overall transformation trend for designing, implementing, deploying, managing and maintaining network equipment and network components by software programming. By exploiting the flexibility and rapidity natures of software, network softwarization targets on consolidating the independent efforts from industry to promote the delivery

of new services with low CAPEX and OPEX [347]. Current network softwarization trend mainly focuses on software defined infrastructure. Due to the typical three-layer structure of programmable network enabled by SDN, we also discuss the software defined control and application respectively in order to give a comprehensive presentation.

7.1.1. Software defined infrastructure

There are a majority of companies including Google, Amazon, IBM and Oracle which have invested in cloud computing and offered individuals and enterprises a variety of cloud-based solutions, for example, social networking (Facebook, Twitter, LinkedIn), email (Hotmail, Gmail, Yahoo email) and other document services (Google Docs, Zoho office) [21]. Due to such massive efforts contributed to cloud computing, and a lot of mature technologies developed, today's world is in a transition to cloud computing which enables customers to consume compute resources as public facilities rather than having to build and maintain computing infrastructure themselves [382]. Despite this, we should be aware that traditional cloud computing architecture may be powerless to support such world-wide transition. Thus, current architecture of data centers should be radically re-designed based on a unified standard in order to comply with such trend. Under this situation, the new paradigm Software Defined Infrastructure (SDI) [348] and [349] offers the opportunity.

The physical infrastructure mainly includes two kinds of entities which are devices and resources. As for SDI, they are virtualized. In particular, the hardware based devices are virtualized as software (e.g., virtual switch and virtual router), and the physical resources correspond to the virtual resources in SDI [280]. Such two kinds of virtual entities can generally constitute a simple virtual network. As a result, the physical network only needs to handle traffic forwarding, while virtual networks are responsible for programming service deployment and managing resources in a smart and automated way [385]. In addition, we can build multiple independent and virtual network environments over the same SDI, thus to provide a multi-tenant environment [384]. Despite this, traffic belonging to different tenants should be isolated in case they mix with each other and affect the regular data forwarding process. Unlike traditional services that are delivered via specialized hardware, SDI abstracts a virtualized environment that logically locates above the physical network, such that the service deployment can be fulfilled automatically and remotely instead of touching the underlying physical infrastructure manually [321].

Generally, SDI intends to integrate the software defined computing, networking and storage into a fully software defined data center, in which the infrastructure can be fully deployed and controlled by software (e.g., applications) without the interference of human [350]. In the SDI based data centers, customers can focus on defining their applications and operation policies, while the requirements on infrastructure provision and configuration are automatically fulfilled by the orchestration software [351]. In legacy data centers, the resources are used in a pay-as-use manner and the amount of resources provided to customers does not change despite the change of service demand [383]. On the contrary, SDI enables a more flexible way for resource allocation by continuously monitoring the network condition. Specifically, once the change of service demand is detected, the orchestrator will intelligently analyze customers' needs and decide in real-time which and how much resource can provide the best support. Then, the orchestrator will automatically adjust the allocated resources to meet the policy requirements [352].

Due to the software based features of SDI, it can be combined with the technology of machine learning. On one hand, the service orchestrator can leverage machine learning to learn from the past experiences and lessons. On the other hand, based on big data, machine learning even offers a chance for service orchestrator to predict the future requirements and make the corresponding preparations [353]. Currently, such machine learning based prediction strategy is mainly used for VNFs. For example, Shi et al. [267] leveraged machine learning to analyze the large amount of data generated by users and to make the best decisions for VNF deployment. By introducing such capabilities, SDI enables data centers with the ability of self-analysis. In this way, more and more projects are developed for studying and building suitable SDI for purposes such as test and production (e.g., [20], [49] and [354]). However, considering the fact that cloud computing is the basics of SDI, most TOs start building their SDIs by combining cloud computing and NFV. For example, CloudNFV [20] is a platform integrating SDN, NFV and cloud computing, which offers a programmable infrastructure to customers. Another example is OpenStack [49] which begins as a cloud platform. Then, it gradually integrates the NFV features and develops essential plugins and APIs for service providers and vendors to manage their infrastructure easily and flexibly.

Although SDI brings a lot of benefits such as automation and flexibility, it also introduces many potential pitfalls and challenges during its implementation and operation. As we know, virtualization is inevitable for legacy systems in the transition to SDI based systems [355]. On one hand, the physical resources (e.g., CPU, memory and storage) are usually provided by isolated purpose-built servers which are maintained independently. Through the virtualization of network resources, we can eliminate such isolated situation, and achieve dramatic cost reduction and agility improvement. On the other hand, the virtualization of network would inevitably generate a large amount of virtualized workloads. In this way, the corresponding management, monitoring and maintenance manner of the overall infrastructure should be altered in order to better satisfy the needs of these virtual workloads [356]. Nevertheless, the current situation is that most enterprises are only able to partially implement the virtualization of their infrastructure, due to reasons such as cost and geographic locations. In this way, the coexistence of consolidated virtual resources and the isolated physical resources would cause many unavoidable problems in terms of cooperation and compatibility, which may be even worse in large scale networks and finally lead to great complexity and cost [357]. In addition, to satisfy the needs of these virtualized workloads and the expectations of users, enterprises have to design their infrastructure environment on a per-service basis, which indicates that different resource allocation models should be developed to serve different services [358]. All these factors contribute to the complexity of software defined infrastructure.

The coexistence between traditional infrastructure and the software defined infrastructure is inevitable and has been around for a long time, which cannot simply be solved by the new features of SDN and NFV. However, maintaining the two environments at the same time is unsustainable due to the increased complexity and cost [359]. In this regard, the concept of composable infrastructure [360] is proposed as a transitional product, which allows operators to deliver new services across the traditional and software defined infrastructures. With such hybrid infrastructure, many limitations can be solved. In particular, on one hand, the customers can use existing traditional applications to support the basic business needs. On the other hand, SDI based applications can be combined with technologies such as big data and cloud-native to further improve network profits and user experiences [361]. Moreover, for the purpose of consolidating efforts from industry, most composable infrastructures have provided a unified API, such that the third-party tools can be integrated with the software defined infrastructure [360]. Therefore, we can regard the composable infrastructure as a kind of 'semi-SDI'.

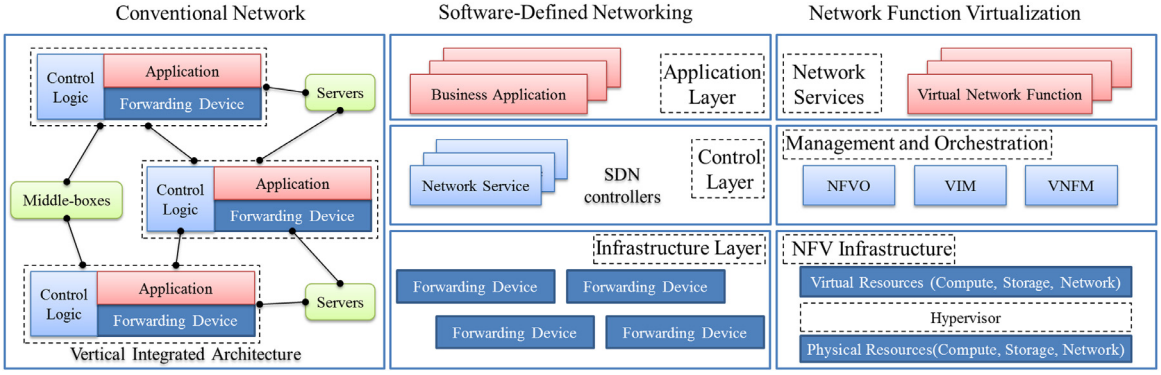

Fig. 14. Basic architecture of conventional network, SDN and NFV.

7.1.2. Software defined control

SDN and NFV are two emerging paradigms designed to enable customers and enterprises with a more agile and cost-effective networking architecture [77], [345]. Their basic structures are shown in Fig. 14, along with the conventional networking architecture. Despite the fact that SDN and NFV are two different ideas in terms of functionality and concept, they are highly complementary [373]. By observing Fig. 14, we can notice that they abstract the control logic from the underlying infrastructure, which enables the programmability of both control plane and data plane. Despite this, it is aware that SDN and NFV are mainly focused on the control plane programmability, such that enterprises and operators can pay more attention on determining their operation and control logics in order to adapt to the dynamically changing requirements [256]. For example, by softwarizing the infrastructure platform, NFV offers a virtualized environment for executing and deploying various SDN entities such as application [374] and controller [375]. Besides, SDN offers a centralized network view and a certain extent of control plane programmability for users to maintain and operate the network toward a better condition [104].

The integration of SDN and NFV introduces an open source control plane with high programmability. On one hand, replacing the proprietary hardware with the commodity server not only decreases the time for service to market, but also builds a scalable and elastic platform for supporting the control plane programmability [376]. On the other hand, separating network functionalities from hardware and running them on commodity servers accelerate network innovation and offer flexible network control and configuration [377]. Taking these factors into consideration, SDN and NFV jointly present a fully software defined control plane which can flexibly determine various network polices to optimize the network performance (e.g., reduction of cost and power consumption). For example, based on the control plane programmability offered by SDN and NFV, Luo et al. [329] carried out a dynamic controller consolidation and sleep mechanism to achieve high energy efficiency, while Kellerer et al. [378] intended to implement a network measurement solution in terms of flexibility, scalability, etc.

As explained, service function chain is one of the most important use cases in NFV. On one hand, from the perspective of SDN, it offers a global network view for service function chaining. On the other hand, the concept of VNF enables a flexible way for service composition. These characteristics reflect the trend of softwarization which is discussed in many literatures such as Refs. [379]–[381]. Particularly, Duan et al. [379] discussed the challenges caused by softwarization, while Gau et al. [380] and Zhang et al. [381] leveraged the benefits of softwarization to maximize the service capacity and service provision scalability respectively.

Due to the control plane programmability enabled by SDN and NFV, we can describe the control logics in software and execute them via controller [362]. For example, the VNF related algorithms discussed in Section 5.4 can be implemented by the controller in order to achieve a better performance for service chaining and provision. From the perspective of service oriented control plane, it has to cover the basic responsibilities that may be demanded by customers, which include software defined topology, software defined resource allocation, and software defined protocol [347]. Specifically, for the software defined topology, it indicates the logical forwarding topology for a given service chain consisting of multiple sequential network functions hosted on different physical network nodes [363]. Nevertheless, the service provision requires not only placing the required functions on physical nodes, but also configuring the network. In this regard, the topology discovery technology should be well studied in order to provide the up-to-date and real-time network condition [364]. For the software defined resource allocation, it operates after the construction of the logical forwarding topology. In particular, it requires to build the mapping relationship between the logical topology and the physical infrastructure in terms of resource (e.g., storage and bandwidth) allocation [173]. In particular, the corresponding resource allocation algorithms have already been discussed in Section 5. For the software defined protocol, it intends to design the protocols that will be used by control plane to communicate with data plane and application plane respectively, thus achieving the network programmability [365]. Apart from establishing the protocol stack, dynamically adjusting the logical functional units according to application type, service quality and physical resource mapping should be taken into consideration when fulfilling the protocol design [199].

7.1.3. Software defined application

SDN and NFV enable the appearance of various kinds of network applications which have different requirements in terms of latency, bandwidth, etc. For example, some applications (e.g., video related) may demand large bandwidth, while other applications (e.g., online meeting) may be sensitive to the network delay. Therefore, a unified application management system is required to accommodate the applications with different requirements [366]. In the context of NFV, the concept of the software defined application also includes various VNFs. In this way, the new features of software defined applications are that they can be dynamically instantiated and executed on the COTS based network according to users' demands [196]. One or more VNFs can be used to constitute new services [239]. However, to create a new service with high quality, the VNF placement and corresponding resource allocation problems should be carefully addressed, which have already been discussed in Section 5.1.

Due to the wide adoption of network virtualization, current application services can be flexibly created and provided [281]. How-

ever, it is aware that the demand of these services may be changed during their life cycles. Such situation is hard to handle in traditional networks that rely on purpose-built hardware. In contrast, the virtualized network enables an easy way for resource reconfiguration and re-allocation, such that the resources allocated to each application can be customized. On one hand, the customized approach increases resource utilization. On the other hand, the resource customization also results in management complexity, because it is carried out based on arbitrary user requirements [367]. Unlike traditional network that offers one kind of application service to multiple customers with the same quality, the software defined application intends to offer differentiated service according to users' preferences and network environments [368]. This objective can be fulfilled by isolating network environments with different virtual resources, where each isolated environment should offer customized resources for each application [369]. Despite this, the management of these isolated and virtualized environments is complex, and necessarily required.

The service customization framework is very important to address the above challenges. One typical example is the Application Driven Network (ADN) [370] proposed by Huawei, which puts forward that network design and evolution should be driven by network applications and usages. In particular, this ADN architecture is fundamentally different from the traditional network architectures, which aims at optimizing network operation and resource usage. Unlike ADN that presents an overall framework for various applications, other researches focus on one particular application. For example, Monshizadeh et al. [371] intended to create a threat detection application for SDN, while Luo et al. [372] focused on providing the context-aware traffic forwarding applications. Nevertheless, the users' demands and the dynamically changing network conditions should be taken into consideration when designing and implementing these applications.

Now, with the advent of 5G and IoT era, the number of end user devices will reach billions [430], which means that the quantity and variety of applications are also going to increase dramatically. In this way, the corresponding challenge is to build an intelligent application management system with the ability to maintain and operate such tremendous amount of applications in accordance with service profiles of use and location. Besides, with so many applications that are software based and open source, they are suffering a series of problems including security, authentication, authorization, etc.

7.2. Softwarization in 5G

Generally speaking, 5G indicates the 5th generation mobile network which brings a significant improvement in the traffic processing speed. Accordingly, high speed means large amount of data. As for 5G, it has to support not only massive network connections, but also an increasingly diverse set of services, applications and end users. Currently, there are already a lot of researches presenting a comprehensive survey about 5G. For example, Tayyaba and Shah [398] and Bouras et al. [399] discussed the challenges and potential issues of integrating 5G network with other popular technologies such as SDN and NFV. Ordonez-Lucena et al. [400] particularly discussed the implementation challenges of network slicing technology in 5G, while Gavrilovska1 et al. [401] and Agiwal et al. [402] focused more on optimizing the 5G network architecture in terms of technical requirements and the potential enablers. Different from these researches, completely surveying 5G is not the main scope of this work. We prefer discussing the potential relationship between 5G and NFV, as well as how NFV promotes 5G in this section.

Actually, due to the highly complementary relationship between NFV and SDN, they are usually used together to benefit 5G in terms of network architecture, service provision, etc. For example, by introducing the classical SDN three-layer structure into the 5G based networks (e.g., mobile network [386], transport network [387] and RAN [388]), a highly centralized environment is achieved, in which the network monitor, configuration and policy enforcement can be carried out easily and flexibly. In addition, by decoupling network functionalities from the hardware, Taleb et al. [390] demonstrated a concept of "anything as a service" which allowed network operators to dynamically create and orchestrate 5G based services on demand. Furthermore, Mechtri et al. [389] jointly introduced SDN, NFV and cloud into 5G networks, and proposed a service oriented network architecture for the purpose of reducing the average service creation time in 5G networks. Therefore, on one hand, enabled by the integration with SDN and NFV, the implementation of 5G becomes feasible and reasonable. On the other hand, benefiting from the features of 5G, many traditional paradigms such as EPC, RAN and even satellite communications have regained high attention.

Nevertheless, in order to embrace the benefits brought by the combination of 5G and SDN/NFV, the most important aspect is the virtualization of various network functions (e.g., the S/P-GW functions in EPC) in different network fields, such that these functions can be managed in a more flexible manner via software. For example, by implementing the virtualization of EPC, Cau et al. [391] regarded the virtual EPC as a kind of service and deployed it over a cloud infrastructure to support elastic service provision. Importantly, due to the integration with SDN, these virtualized functions are usually managed by the centralized controller. For example, Ksentini, and Nikaein [392] relied on the SDN controller to fulfill the policy enforcement for virtual 5G enabled functions. It is well known that the centralized control suffers from scalability issue. In this case, Aissioui et al. [393] applied the three-layer structure of SDN into the 5G supported cloud system, and separated the control plane into global and local controllers for distributed management, while Vilalta et al. [394] proposed to take charge of heterogeneous domains with different controllers. C-RAN [174], the integration of RAN and cloud, can better adapt to the growing bandwidth needs and various traffic patterns by centralizing the control of wireless functions that run on COTS based hardware. Nevertheless, in order to achieve benefits such as flexible mobility and high quality, 5G technology is inevitable for C-RAN. For example, Ref. [395] achieved flexible service-tailored mobility, service-aware control and network-aware orchestration by using the network slicing technology enabled by 5G. With respect to the field of satellite communication, 5G technology also brings a lot of opportunities. However, due to the reliability consideration, the virtualization happens in the satellite ground segment system. For example, both Ref. [396] and Yousaf et al. [397] leveraged the concepts of SDN and NFV to build an architecture for applying 5G into the satellite ground segment system, in which the integration with SDN and NFV is the key facilitator to better deliver the satellite communication services.

By observing and analyzing all these related work, we can discover a trend that the network is becoming more and more softwarization under the driven of NFV and SDN. As for 5G, it also benefits from such softwarization and obtains a major capability, that is, network slicing which actually reflects the trend of network softwarization [346]. The concept of network slicing in 5G usually refers to an independent connection of virtual resources and functions that are implemented and managed by software [403]. The resources and functions in this slicing are used to satisfy customers' requirements in terms of service quality, reliability, etc. The key feature of network slicing is that it allows the coexistence of multiple logically isolated network partitions over the same infrastructure [404]. In particular, the Next Generation Mobile Networks (NGMN) has published a white paper to identify the relationship

between network slicing and 5G, which intended to embrace the benefits (e.g., flexible and fast service provision) enabled by SDN and NFV [405]. Nevertheless, due to the fact that the 5G technology is going to be used for a large variety of applications, it is essential to extend this network slicing concept to a wide range of use cases especially those associated with SDN and NFV [406]. However, there are still some major issues should be addressed before achieving such goal.

As explained, the network slicing is a key concept in 5G. How to create network slices on top of infrastructure and manage them should be carefully addressed. First of all, network slices are usually used to serve independent elements. For example, one network slice can be used to provide an isolated environment for VNFs [407]. However, due to the real-world requirements for 5G, the network slicing should be extended to support end-to-end services or communications which include the consideration for software defined infrastructure and the entire service function path [408]. Currently, the SDN technology can be used for the software defined infrastructure. However, SDN is primarily proposed for a particular set of network scenarios such as data center and transport networks, while 5G is usually applied to wireless network scenarios. In this way, there exists a gap between current SDN technology and the requirement for end-to-end quality in 5G [409]. It is desired that an infrastructure for 5G will support end-to-end control and management of slices and the composition of multiple slices, especially with the consideration of slicing over wireless and wired parts of end-to-end paths. In addition, due to the expectation for 5G, it may be used to support various communication protocols (including those that have not yet appeared) in the near future [410]. To fulfill this target, infrastructure should be enabled with data plane programmability, namely, software defined infrastructure. Directly applying SDN for data plane programmability is not expected, because SDN primarily focuses on the softwarization of control plane [256]. Despite this, we can combine SDN and some other tools to jointly fulfill this purpose. For example, P4 [96] is a programming language designed to allow the data plane programmability, and Protocol Oblivious Forwarding (POF) [411] discusses the potential of extending SDN to support new forwarding protocols. Through such combination, we may build the infrastructure for 5G with deep data plane programmability.

Due to the sweeping trend of network softwarization, most operators would like to softwarize everything in the network, thus to meet various network management and service objectives as many as possible [346]. However, it is aware that not every component of the infrastructure may be defined by software and made programmable, considering the trade-off between programmability and performance [213]. As for 5G, such case is more obvious, because some applications in 5G have stringent performance requirements such as low latency and high throughput [412]. In this way, the infrastructure of 5G may need to support traffic classification performed not only by flow-basis, but also by other metrics and bundles such as per-device and per-application basis so as to apply software/hardware based solutions appropriately tailored for individual use cases [413]. As explained, the high performance is typically guaranteed by dedicated hardware or hardware acceleration technologies, while the programmability is offered by software. Therefore, it is necessary to clearly define the role of hardware and software according to the potential use cases when softwarizing infrastructure of 5G.

Currently, the standardization work around 5G is in progress by many world-known telecommunication consortiums like 3GPP, while there are also some SDOs (e.g., NGMN and 5GMF) particularly focusing on the 5G softwarization specifications. However, most of those standards are not yet solidified. Besides, many well-known companies, such as the Verzon and AT&T, are already testing 5G. Still, they do not announce any actual achievement to the public. Thus, most experts even forecast that 5G will not be widely available until 2020 [414]. The support of inter-operability and compatibility with legacy and non-virtualized network functions is not yet investigated in 5G networks. All these lead to the fact that 5G is in the infancy. Fortunately, NFV and SDN, as two promising paradigms, are expected to accelerate the standardization process of 5G. The pressure of the explosive amount of traffic generated by worldwide mobile devices can be eased by the joint architecture of SDN and NFV [401]. However, what concerned us is that the traffic grows exponentially especially in high speed 5G networks. Once the amount of the traffic reaches a certain threshold, how to guarantee the performance is strongly worth thinking.

7.3. Softwarization in IoT

The Internet of Things (IoT) indicates a system that connects diverse devices (e.g., phone, vehicle, sensor, etc.) to the Internet. Each device has a unique identifier (currently IP addresses) to communicate with each other and the ability to transfer data over the Internet without requiring human-to-human or human-to-computer interactions [427]. However, considering the heterogeneous structure of IoT, the network connectivity among these devices should be flexible and reliable across many scenarios such as homes and offices, because they need connectivity with not only other devices, but also the non-IoT world [428]. Currently, instead of using hardware based solution (e.g., VLAN) to manage access to these devices, the solution (e.g., [65] and [66]) offered by SDN and NFV is a better option, which leverages a centralized controller to carry out customers' plans on remote devices in terms of routing, resource allocation, etc.

Besides, among these diverse and dense IoT devises, there are a lot of them offering universal network functions such as firewall, traffic engineering and load balancing. In particular, such network functionalities are provided based on proprietary hardware. Once new requirements appear, a new generation of more sophisticated and costly hardware will be required to replace the old dedicated one [24]. In contrast, the network softwarization offered by SDN and NFV can bring hardware independence by decoupling network functionalities from hardware and implementing them as software. In this way, the new network or service requirements can be fulfilled with a simple software update [429]. However, since the number of devices connected to the Internet is estimated to reach billion by 2020 [430], the centralized management and control approach offered by SDN is obviously not applicable. In this aspect, distributed approaches are more promising. For example, Abeele et al. [442] proposed a distributed and intelligent architecture to handle the hyper-scale data generated by these IoT devices with the sensor functions virtualized, while Yigitoglu et al. [431] presented a distributed IoT orchestration architecture which enabled an intelligent partition of a real time IoT computing task into an optimal coordination of server-side processing and IoT object side processing.

The integration with SDN and NFV offers great network agility for managing tremendous amount of end devices in IoT. Specifically, on one hand, NFV introduces the concept of VNF which can be adopted to deliver the customized network services demanded by IoT. On the other hand, SDN offers a centralized way for managing and orchestrating flows over distributed IoT networks, which is more flexible and cost-saving. In this regard, a lot of research work from both industry and academia have been carried out. For example, the HP enterprise has highlighted the positive effects that SDN and NFV may have on IoT, which are published in a business white paper [439]. As explained, programmability and virtualization are two main features enabled by SDN and NFV. Based on this, Omnes et al. [12] built a programmable and virtualized network infrastructure for IoT, which was expected to handle some identi-

fied IoT challenges, while Du et al. [441] intended to build a software defined IoT data plane. Although they both focused on creating a software defined infrastructure for IoT, they were actually different, because Ref. [12] was service-aware and Ref. [441] was context-aware. In addition, SDN and NFV also offer opportunities for applying IoT to other fields. For example, based on the network softwarization enabled by SDN and NFV, Salahuddin et al. [444] built an IoT system for smart healthcare applications and services, while Hegyi et al. [445] applied the edge cloud benefits to the deployment of IoT. Furthermore, a detailed survey on applying SDN and NFV to IoT could be found in Ref. [446] which outlined the ways for SDN and NFV combination, and reviewed some general SDN/NFV enabled IoT architectures, along with the real-life deployments and use cases.

IoT actually means tremendous devices continuously generating and exchanging massive data. On one hand, such enormous data can be replicated a lot, which means a lot of redundant information and results in high network burden [432]. On the other hand, these data usually require to be handled in real time [433]. However, due to the latency and bandwidth limitation, not all of them can be analyzed in time. Current approaches used to achieve the real time operations are generally based on edge computing to cache contents at the network edge for local information processing or making them available for subsequent requests from the same domain [440]. In this aspect, SDN offers a global network view to help the cache deployment, while NFV offers a flexible implementation for in-network caching functions. Despite the benefits enabled by SDN and NFV, we should be aware that their inherent limitations are suffered by IoT due to the integration [440]. On one hand, the most obvious limitation of SDN is the network performance bottleneck caused by the centralized controller, which has already been explained. On the other hand, the function virtualization enabled by NFV brings challenges as well as benefits. For example, by packing the virtualized functions in a portfolio, Hernando et al. [443] offered an inexpensive solution for supporting most IoT physical communication options. However, this operation actually exposes a virtualization domain to the attackers, which may lead to security and privacy problems.

Network softwarization enabled by SDN and NFV is currently being accelerated by many technical and economic drivers such as increased performance of processing and storage at continuously decreasing costs [434]. This trend will have a huge impact on reshaping current IoT ecosystems and creating new opportunities, because softwarization will gradually and inevitably eliminate the boundary between the Internet and the elements connected to it [435]. In such situation, more and more powerful IoT devices (e.g., user terminal, machine, robot, etc.) will perform like network nodes that store data locally and even execute network functions and services [436]. Therefore, the implementation of softwarization at the edge network and IoT will constitute a borderless platform with logical resources, which is fully decoupled from the underlying physical infrastructure and spanned across devices, network nodes, and up to the cloud. Multiple services can be dynamically created and provided through this platform. From this perspective, network softwarization benefits not only IoT, but also other technologies such as big data and digital money, thus creating a huge wave of innovation across all industries and paving the way towards a digital society [437].

7.4. Softwarization in ICN

Information-Centric Networking (ICN) is an emerging network paradigm proposed to address the huge mismatch between the rapid growing content (e.g., audio or video) requests and the host-centric network model [438]. The main feature of ICN is that it defines a unique name for each content, such that the data can be accessed by the name instead of IP address. Since the in-network caching is used in ICN, users can get the required content from the nearest content holders directly [415]. In this way, the content in ICN becomes independent from the location, storage and even transportation, which enhances many features such as security and mobility. Based on this, many independent frameworks for ICN were proposed, for example, Li et al. [416] and Nishiyama et al. [417] which indeed accorded with the characteristics of ICN and addressed many issues. Despite this, there are still many issues that cannot be solved by these frameworks, for example, the tremendous amount of content names. In fact, using the content name to search data is a new data plane forwarding mechanism different from the IP based one in Internet. In this regard, the data plane programmability of ICN should be enabled.

Currently, the integration between ICN and SDN/NFV is widely recognized, because they can enable ICN with a certain extent of data plane programmability and network virtualization. For example, by virtualizing the edge functions, Ravindran et al. [418] proposed an ICN based edge cloud framework which supported SDN, NFV, ICN protocols and exploited ICN features (e.g., name based routing and caching) to provide real time and non-real time services. In addition, different ICN architectures are designed to satisfy various requirements. Due to the data plane programmability and virtualization enabled by SDN and NFV, we can deploy multiple ICN architectures over the same physical infrastructure. For example, in order to fully support the ICN features, Ren et al. [419] built a unified framework with such two capabilities enabled, while Trajano and Fernandez [420] allowed network orchestration and high scalability at runtime based on SDN and NFV. In this situation, both of them were able to dynamically adjust the content caching according to business needs. Salsano et al. [421] introduced a concept of generalized virtual networking which offered a framework to influence the routing of packets based on service level information carried with the packets. Such framework was based on a protocol header inserted between network and transport layers, and thus it was seen as a layer 3.5 solution. Based on such design, this framework demonstrated to support ICN, NFV and SDN. Ueda et al. [422] integrated the name look-up principle of ICN into the software switches (e.g., Open vSwitch) supported by virtualized infrastructure. Based on this, the ICN packets could be forwarded without the long prefix match searching and the computational overhead was also mitigated.

Overall, the integration with SDN and NFV brings many benefits to ICN. Likewise, ICN introduces a new way for service provision in SDN and NFV. For example, by regarding the virtualized network functions as the content, it is easy to determine where to place the required network functions precisely. Based on this, Arumaithurai et al. [423] proposed a function centric service chain solution which could react to failures with fewer packet loss and adapt to new network functions quickly. In addition, based on the characteristics of ICN, services can be separated from their locations in the network. Arumaithurai et al. [424] leveraged this feature of ICN to enhance the service composition and provision in SDN and NFV, which not only enabled a dynamic instantiation of network functions, but also resulted in efficient, scalable and reliable service provision. Besides, through data plane programmability and network virtualization, operators can take part in the process of end-to-end service delivery in terms of service composition, orchestration, routing and resource allocation.

Despite the benefits of SDN and NFV, many innovative network functions (e.g., name based routing) in ICN are not supported by current network devices. Targeting on this point, the programmability and virtualization capabilities can actually be used for the real implementation of ICN. For example, Ren et al. [425] proposed a novel ICN deployment framework by using a lot of software components, while Ravindran et al. [426] built a general, flexible and

application-driven framework for ICN by using the network slicing concept in 5G. In addition, it is worth noting that ICN introduces the named data as the network primitive. Thus, the amount of content names would increase explosively (may follow exponential distribution) with the development of ICN. As a result, how to deal with such tremendous amount of names must be figured out.

8. Conclusion

The network is becoming more and more ossified and inflexible with the increasing amount of middle-boxes. NFV has been proposed to eliminate such situation by decoupling network functions from the proprietary hardware. Such decoupling enables a strong, scalable and elastic ecosystem, in which the network management and orchestration are automated. In this case, the network operators and service providers can run software based functions (i.e., VNFs) on COTS based servers instead of purpose-built hardware, which actually reduces CAPEX and OPEX. In addition, considering the complementary relationship between SDN and NFV, more and more researches are focusing on applying this integrated architecture into other scenarios such as 5G and IoT. Therefore, it is promising that the NFV and SDN integrated architecture would lead the networking trend (i.e., softwarization) in the near future.

In this paper, we present a comprehensive survey of NFV which includes three big parts, that is, the basic concepts of NFV (including motivation, terminologies, standardization efforts, history and architecture), VNF related algorithms (including VNF placement, scheduling, migration, chaining and multicast), challenges and future directions. In particular, the NFV architecture is introduced using a bottom up approach in order to highlight its hierarchical structure. Besides, the combination between NFV and SDN has enabled a trend towards network softwarization. Thus, the future direction is particularly discussed around such softwarization topic in order to drive NFV moving forward. Nevertheless, NFV is still in the infancy. Therefore, we will continuously focus on the extensive activity around NFV in the near future as there are so many new topics to be explored, for instance, the migration path to NFV, NFV inter-operability and service composition.

Acknowledgment

This work is supported by the Major International(Regional) Joint Research Project of NSFC under Grant No. 71620107003, the National Natural Science Foundation of China under Grant No. 61572123, the National Science Foundation for Distinguished Young Scholars of China under Grant No. 71325002, the Foundation for Innovative Research Groups of National Science Foundation of China under Grant No. 61621004, and the MoE and ChinaMobile Joint Research Fund under No. MCM20160201.

References

[1] Z.A. Qazi, C.-C. Tu, L. Chian, R. Miao, V. Sekar, M. Yu, SIMPLE-fying middlebox policy enforcement using SDN, in: Proceedings of ACM SIGCOMM 2013, 43, Hong Kong, China, 2013, pp. 27–38, doi:10.1145/2534169.2486022.

[2] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, V. Sekar, Making middleboxes someone else's problem: Network processing as a cloud service, in: Proceedings of ACM SIGCOMM 2012, 42, Helsinki, Finland, 2012, pp. 13–24, doi:10.1145/2342356.2342359.

[3] A.B. Barr, Y. Harchol, D. Hay, Openbox: Enabling innovation in middlebox applications, in: Proceedings of the 2015 ACM SIGCOMM workshop on Hot Topics in Middleboxes and Network Function Virtualization, 2015, pp. 67–72. London, UK, doi: 10.1145/2785989.2785992.

[4] H. Ko, G. Lee, I. Jang, S. Pack, Optimal middlebox function placement in virtualized evolved packet core systems, in: 17th IEEE Asia-Pacific Network operations and Management Symposium (APNOMS), 2015, pp. 511–514. Busan, doi: 10.1109/APNOMS.2015.7275380.

[5] ETSI, Network function virtualisation-white paper1, SDN and openflow world congress, 2012, Darmstadt, Germany, [Online]. Available: http://portal.etsi.org/NFV/NFV_White_Paper.pdf.

[6] ETSI, Network function virtualisation-white paper2, SDN and openflow world congress, 2013, Frankfurt, Germany, [Online]. Available: http://portal.etsi.org/NFV/NFV_White_Paper2.pdf.

[7] ETSI, Network function virtualisation-white paper3, SDN and openflow world congress, 2014, Dusseldorf, Germany, [Online]. Available: http://portal.etsi.org/NFV/NFV_White_Paper3.pdf.

[8] I. Cerrato, et al., Toward dynamic virtualized network services in telecom operator networks, Elsevier Comput. Netw. 92 (2) (2015) 380–395, doi:10.1016/j.comnet.2015.09.028.

[9] P. Wang, J. Lan, X. Zhang, Y. Hu, S. Chen, Dynamic function composition for network service chain: model and optimization, Elsevier Comput. Netw. 92 (2) (2015) 408–418, doi:10.1016/j.comnet.2015.07.020.

[10] R. Munoz, R. Vilalta, R. Casellas, R. Martłnez, T. Szyrkowiec, A. Autenrieth, V. Lpez, D. Lpez, SDN/NFV orchestration for dynamic deployment of virtual SDN controllers as VNF for multi-tenant optical networks, in: 2015 Optical Fiber Communications Conference and Exhibition (OFC), Los Angeles, CA, 2015, pp. 1–3, doi:10.1364/OFC.2015.W4J.5.

[11] R. Nejabati, S. Peng, M. Channegowda, B. Guo, D. Simeonidou, SDN and NFV convergence a technology enabler for abstracting and virtualising hardware and control of optical networks, in: 2015 Optical Fiber Communications Conference and Exhibition (OFC), Los Angeles, CA, 2015, pp. 1–3, doi:10.1364/OFC.2015.W4J.6.

[12] N. Omnes, M. Boulillon, G. Fromentoux, O.L. Grand, A programmable and virtualized network & IT infrastructure for the internet of things, in: 18th International Conference on Intelligence in Next Generation Networks (ICIN), Paris, 2015, pp. 64–69, doi:10.1109/ICIN.2015.7073808.

[13] I.F. Akyildiz, S.-C. Lin, P. Wang, Wireless software-defined networks (w-SDNs) and network function virtualization (NFV) for 5g cellular systems: An overview and qualitative evaluation, Elsevier Comput. Netw. 93 (1) (2015) 66–79, doi:10.1016/j.comnet.2015.10.013.

[14] F. Yang, A flexible three clouds 5g mobile network architecture based on NFV & SDN, China Commun. 12 (0) (2015) 121–131, doi:10.1109/CC.2015.7386160.

[15] H. Hawilo, A. Shami, M. Mirahmadi, R. Asal, NFV: State of the art, challenges and implementation in next generation mobile networks (vEPC), IEEE Netw. 28 (6) (2014) 18–26, doi:10.1109/MNET.2014.6963800.

[16] E. Haleplidis, S. Denazis, O. Koufopavlou, J. Martin, J.H. Salim, K. Pentikousis, ForCES applicability to SDN-enabled NFV, in: Third European Workshop on Software Defined Networks (EWSDN), 2014, pp. 43–48. Budapest, doi: 10.1109/EWSDN.2014.27.

[17] J. Matias, J. Garay, N. Toledo, J. Unzilla, E. Jacob, Toward an SDN-enabled NFV architecture, IEEE Commun. Mag. 53 (4) (2015) 187–193, doi:10.1109/MCOM.2015.7081093.

[18] W. Ding, W. Qi, J. Wang, B. Chen, OpenSCaas: An open service chain as a service platform toward the integration of SDN and NFV, IEEE Netw. 29 (3) (2015) 30–35, doi:10.1109/MNET.2015.7113222.

[19] L.I.B. Lpez, L.V. Caraguay, L.J.G. Villalba, D. Lpez, Trends on virtualization with software defined networking and network function virtualization, IET Netw. 4 (5) (2015) 255–263, doi:10.1049/iet-net.2014.0117.

[20] 6Wind, et al., CloudNFV, 2013, [Oline]. Available: http://www.cloudnfv.com/WhitePaper.pdf.

[21] B. Wang, Z. Qi, R. Ma, H. Guan, A.V. Vasilakos, A survey on data center networking for cloud computing, Elsevier Comput. Netw. 91 (0) (2015) 528–547, doi:10.1016/j.comnet.2015.08.040.

[22] N. Figueira, R. Krishnan, L. Diego, S. Wright, Policy architecture and framework for NFV infrastructures, IRTF NFV Research Group (2015) 1–20.

[23] S. Clayman, E. Maini, A. Galis, A. Manzalini, N. Mazzocca, The dynamic placement of virtual network functions, in: IEEE Network Operations and Management Symposium (NOMS), 2014, pp. 1–9. Krakow, doi: 10.1109/NOMS.2014.6838412.

[24] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F.D. Turck, R. Boutaba, Network function virtualization: State-of-the-art and research challenges, IEEE Commun. Surv. Tutor. 18 (1) (2016) 236–262, doi:10.1109/COMST.2015.2477041.

[25] B. Han, V. Gopalakrishnan, L. Ji, S. Lee, Network function virtualization: challenges and opportunities for innovations, IEEE Commun. Mag. 53 (2) (2015) 90–97, doi:10.1109/MCOM.2015.7045396.

[26] R. Mijumbi, J. Serrat, J.-L. Gorricho, S. Latre, M. Charalambides, L. Diego, Management and orchestration challenges in network functions virtualization, IEEE Commun. Mag. 54 (1) (2016) 98–105, doi:10.1109/MCOM.2016.7378433.

[27] S. Abdelwahab, B. Hamdaoui, M. Guizani, T. Znati, Network function virtualization in 5g, IEEE Commun. Mag. 54 (4) (2016) 84–91, doi:10.1109/MCOM.2016.7452271.

[28] L.M. Contreras, P. Doolan, H. Lonsethagen, D.R. Lpez, Operational, organizational and business challenges for network operators in the context of SDN and NFV, Elsevier Comput. Netw. 92 (2) (2015) 211–217, doi:10.1016/j.comnet.2015.07.016.

[29] K. Edeline, B. Donnet, Towards a middlebox policy taxonomy: path impairments, in: 17th IEEE International Workshop on Network Science for Communication Networks (NetSciCom 2015), 2015, pp. 402–407. Hong Kong, China, doi: 10.1109/INFCOMW.2015.7179418.

[30] B. Carpenter, S. Brim, Middleboxes: Taxonomy and issues, RFC 3234, 2002, 1-27.

[31] R. Tu, X. Wang, J. Zhao, Y. Yang, L. Shi, T. Wolf, Design of a load-balancing middlebox based on SDN for data centers, in: IEEE Conference on Computer Communication Workshops, 2015, pp. 480–485. Hong Kong, China, doi: 10.1109/INFCOMW.2015.7179431.

[32] C.-C. Yeh, C.-W. Chiu, Mint: A cost-effective network address translation ar-

chitecture with multiple inexpensive NAT servers, in: IEEE 17th International Conference on Ubiquitous and Future Networks (ICUFN), 2015, pp. 879–881. Sapporo, doi: 10.1109/ICUFN.2015.7182670.
[33] L. Vdovin, P. Likin, A. Vilchinskii, Network utilization optimizer for SD-WAN, in: 2014 IEEE International Science and Technology Conference, 2014, pp. 1–4. Moscow, doi: 10.1109/MoNeTeC.2014.6995604.
[34] A.J. Jabir, S. Shamala, Z. Zuriati, N. Hamid, A comprehensive survey of the current trends and extensions for the proxy mobile IPv6 protocol, IEEE Syst. J. 99 (0) (2015) 1–17, doi:10.1109/JSYST.2015.2497146.
[35] R.K. Sharma, H.K. Kalita, B. Issac, Different firewall techniques: a survey, in: 5th International Conference on Computing, Communication and Networking Technologies (ICCCNT 2014), 2014, pp. 1–6. Heifei, China, doi: 10.1109/ICCCNT.2014.6963102.
[36] A. Chudnov, D.A. Naumann, Inlined information flow monitoring for javascript, in: ACM SIGSAC Conference on Computer and Communications Security, 2015, pp. 629–643. Denver, Colorado, USA, doi: 10.1145/2810103.2813684.
[37] W. Bulajoul, A. James, M. Pannu, Improving network intrusion detection system performance through quality of service configuration and parallel technology, Elsevier J. Comput. Syst. Sci. 81 (6) (2014) 981–999, doi:10.1016/j.jcss.2014.12.012.
[38] A. Saad, A.R. Amran, I. Norkhalim, M.A.M. Yusof, Automated intrusion detection and prevention system over SPIT (AIDPos), in: International Symposium on Technology Management and Emerging Technologies (ISTMET), 2015, pp. 58–63. Langkawai island, doi: 10.1109/ISTMET.2015.7359001.
[39] S.S. Shankar, P. Lin, A. Herkersdorf, Deep packet inspection in residential gateways and routers: Issues and challenges, in: 14th International Symposium on Integrated Circuits (ISIC), 2014, pp. 560–563. Singapore, doi: 10.1109/ISICIR.2014.7029481.
[40] V. Sekar, N. Egi, S. Ratnasamy, M.K. Reiter, G. Shi, Design and implementation of a consolidated middlebox architecture, in: 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2012, pp. 323–336. San Jose, CA, doi: 10.1145/1096536.1096541.
[41] J. Sherry, Future Architectures for Middlebox Processing Services on the Internet and in the Cloud, EECS Department, University of California, Berkeley, 2012 Master's thesis.
[42] J. Lee, J. Tourrilhes, P. Sharma, S. Banerjee, No more middlebox: integrate processing into network, ACM SIGCOMM Comput. Commun. Rev. 40 (4) (2010) 459–460, doi:10.1145/1851275.1851262.
[43] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. Morris, S. Shenker, Middleboxes no longer considered harmful, in: 6th Symposium on Operating Systems Design and Implementation, 2004, pp. 215–230. doi: 10.1145/606272.606299.
[44] A. Gember, et al., Stratos: A Network-Aware Orchestration Layer for Virtual Middleboxes in Clouds, Cornell University Library, 2013, p. 1-14.
[45] E. Kohler, R. Morris, B. Chen, J. Jannotti, M.F. Kaashoek, The click modular router, in: Proceedings of the 17th Symposium on Operating Systems Principles, 1999, pp. 217–231, doi:10.1145/319344.319166.
[46] M.-A. Kourtis, et al., Enhancing VNF performance by exploiting SR-IOV and DPDK packet processing acceleration, in: 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), 2015, pp. 74–78. San Francisco, CA, 10.1109/NFV-SDN.2015.7387409.
[47] L. Rizzo, Netmap: a novel framework for fast packet i/o, in: 2012 USENIX Annual Technical Conference, Boston, MA, 2012, doi:10.1145/774763.774771.
[48] M. Berman, et al., GENI: A federated testbed for innovative network experiments, Elsevier Computer Networks, special issue on Future Internet Testbeds 61 (0) (2014) 5–23, doi:10.1016/j.bjp.2013.12.037.
[49] O. Ocata, 2017, [Oline]. Available: https://www.openstack.org/.
[50] M.-K. Shin, et al., Verification for NFV-enabled network services, in: IEEE International Conference on Information and Communication Technology Convengence (ICTC), Jeju, 2015, pp. 810–815, doi:10.1109/ICTC.2015.7354672.
[51] H. Woesner, D. Verbeiren, SDN and NFV in telecommunication network migration, in: IEEE 4th European Workshop on Software Defined Networks (EWSDN), Bilbao, 2015, pp. 125–126, doi:10.1109/EWSDN.2015.80.
[52] L. Foundation, Open platform for NFV, 2014, [Online]. Available: https://www.opnfv.org.
[53] ETSI, NFV: Terminology for main concepts, 2014, [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV/001_099/003/01.02.01_60/gs_NFV003v010201p.pdf.
[54] ETSI, Network function virtualization (NFV): Virtual network functions architecture, 2014, [Oline]. Available: http://www.etsi.org/deliver/etsi_gs/NFV-SWA/001_099/001/01.01.01_60/gs_nfv-swa001v010101p.pdf.
[55] ETSI, Network function virtualisation (NFV): Management and orchestration, 2014, [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf.
[56] ETSI, NFV: Architecture framework, 2014, [Online]. Available: http://www.etsi.org/deliver/etsi_gs/nfv/001_099/002/01.02.01_60/gs_nfv002v010201p.pdf.
[57] ETSI, NFV: Infrastructure overview, 2014, [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/001/01.01.01_60/gs_NFV-INF001v010101p.pdf.
[58] ETSI, NFV: Infrastructure-compute domain, 2014, [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/003/01.01.01_60/gs_NFV-INF003v010101p.pdf.
[59] ETSI, NFV: Infrastructure-hypervisor domain, 2014, [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/004/01.01.01_60/gs_NFV-INF004v010101p.pdf.
[60] ETSI, NFV: Infrastructure-network domain, 2014, [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/005/01.01.01_60/gs_NFV-INF005v010101p.pdf.
[61] ETSI, NFV: Service quality metrics, 2014, [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/010/01.01.01_60/gs_NFV-INF010v010101p.pdf.
[62] ETSI, NFV: Resiliency requirements, 2014, [Online]. Available: http://www.etsi.org/deliver/etsi_gs/NFV-REL/001_099/001/01.01.01_60/gs_nfv-rel001v010101p.pdf.
[63] ETSI, NFV: Security and trust guidance, 2014, [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/003/01.01.01_60/gs_NFV-SEC003v010101p.pdf.
[64] ETSI, Open source management and orchestration, 2016 [Online]. Available: https://osm.etsi.org/.
[65] ONF, Openflow-enabled SDN and network function virtualization, 2014, [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/solution-briefs/sb-sdn-nvf-solution.pdf.
[66] ONF, TR-518 relationship of SDN and NFV, 2015. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/onf2015.310_Architectural_comparison.08-2.pdf.
[67] IRTF, Network function virtualization research group (NFVRG), 2015, [Online]. Available: https://irtf.org/nfvrg.
[68] IETF, Service function chaining working group, 2015, [Online]. Available: https://datatracker.ietf.org/wg/sfc/charter/.
[69] S. Sahhaf, et al., Scalable architecture for service function chain orchestartion, in: IEEE 4th European Workshop on Software Defined Networks, Bilbao, 2015, pp. 19–24, doi:10.1109/EWSDN.2015.55.
[70] O. vSwitch, 2013, [Online]. Available: http://vswitch.org.
[71] L. KVM. [Online]. Available: http://www.linux-kvm.org/page/Main_Page.
[72] ATIS, Network functions virtualization forum, 2015. [Online]. Available: http://www.atis.org/NFV/index.asp.
[73] T.B. Forum, 2015, [Online]. Available: https://www.broadband-forum.org/.
[74] T.B. Forum, Technical work in progress, 2015, [Online]. Available: https://www.broadband-forum.org/technical/technicalwip.php.
[75] D.M.T.F. (DMTF), Open virtualization format (OVF), 2015, [Online]. Available: http://www.dmtf.org/standards/ovf.
[76] The 3rd generation partnership project (3GPP), 2015. [Online]. Available: http://www.3gpp.org/about-3gpp/about-3gpp.
[77] ONF, Software-defined networking: The new norm for networks, 2012, [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf.
[78] C. Strachey, Time sharing in large fast computers, in: Proceedings of the IFIP Congress, 1959, pp. 336–341. doi: 10.12691/jcsa-2-3-1.
[79] S. Nanda, A Survey on Virtualization Technologies, ECSL Publishing, 2015, p. 1-42.
[80] J. Sahoo, S. Mohapatra, R. Lath, Virtualization: a survey on concepts, taxonomy and associated security issues, in: Proceedings of the Second International Conference on Computer and Network Technology (ICCNT 2010), Bangkok, 2010, pp. 222–226, doi:10.1109/ICCNT.2010.49.
[81] Citrix, Xenserver, [online]. Available: https://www.citrix.com/products/xenserver/overview.html.
[82] Microsoft, Hyper-v, [online]. Available: http://www.microsoft.com/en-us/server-cloud/solutions/virtualization.aspx.
[83] VMware, vsphere, [online]. Available: https://www.vmware.com/products/vsphere.
[84] ETSI, NFV announcement on work progress, release 2 and the definition of release 3, 2017, [Online]. Available: https://docbox.etsi.org/ISG/NFV/Open/Other/NFV(16)000338_ETSI_NFV_Announcement_on_work_progress-Release_2_and_the_Definition_of_Release_3.pdf.
[85] ETSI, ETSI plugtests report, 2017, [Online]. Available: https://docbox.etsi.org/ISG/NFV/Open/Other/NFVTST(17)000054_1st_NFV_Plugtests_Test_Report.pdf.
[86] ETSI, NFV announcement after NFV#17: work progress and release 2 and release 3 updates, 2017, [Online]. Available: https://docbox.etsi.org/ISG/NFV/Open/Other/NFV(17)000081_ETSI_NFV_Announcement_post_NFV_17.pdf.
[87] Electronic industries association, Define: EIA-310, [Online]. Available: https://www.server-racks.com/eia-310.html.
[88] C. Bu, X. Wang, M. Huang, K. Li, SDNFV-based dynamic network function deployment: Model and mechanism, IEEE Commun. Lett. 22 (1) (2018) 93–96, doi:10.1109/LCOMM.2017.2654443.
[89] T. Ono, Y. Konishi, T. Tanimoto, N. Iwamatsu, T. Miyoshi, J. Tanaka, FlexDAS: a flexible direct attached storage for i/o intensive applications, in: Proceedings of the 2014 IEEE International Conference on Big Data (Big Data), Washington, DC, 2014, pp. 147–152, doi:10.1109/BigData.2014.7004224.
[90] T. Huang, D. Chang, TESA: a temporal and spatial information aware write-back policy for home network-attached storage devices, IEEE Trans. Consum. Electron. 59 (1) (2013) 122–129, doi:10.1109/TCE.2013.6490250.
[91] G. Reitman, Streaming video with storage area networks, SMPTE J. 110 (8) (2001) 517–522, doi:10.5594/J17762.
[92] IBM, System networking rackswitch g8052, 2013. [Online]. Available: http://www-01.ibm.com/support/docview.wss?uid=isg3T7000287&aid=1.
[93] Juniper, QFX5100 ethernet switch, [Online]. Available: http://www.juniper.net/assets/jp/jp/local/pdf/datasheets/1000480-en.pdf.
[94] A.B. Siddique, K.N. Prabhu, Replacement of heat sink fan by nanocoolants for enhancement of CPU efficiency, in: National Conference on Challenges in Research & Technology in the Coming Decades (CRT 2013), Ujire, India, 2013, pp. 1–7, doi:10.1049/cp.2013.2553.

[95] Oracle, The value of full hardware offload in a converged ethernet environment, 2012, [Online]. Available: http://www.oracle.com/us/products/full-hardware-offload-367052.pdf.
[96] P. Bosshart, et al., P4: programming protocol-independent packet processors, in: ACM SIGCOMM, 44, 2014, pp. 87–95, doi:10.1145/2656877.2656890.
[97] Z. Wang, C. Wu, M. Grace, X. Jiang, Isolating commodity hosted hypervisors with hyperlock, in: Proceedings of the 7th ACM European Conference on Computer Systems, Bern, Switzerland, 2012, pp. 127–140, doi:10.1145/2168836.2168850.
[98] R. Johnston, et al., Xen network flow analysis for intrusion detection, in: Proceedings of the 11th Annual Cyber and Information Security Research Conference, Oak Ridge, TN, USA, 2016, doi:10.1145/2897795.2897802.
[99] R. Sherwood, et al., Flowvisor: a Network Virtualization Layer, 2009, [Online]. Available: http://OpenFlowSwitch.org/downloads/technicalreports/openflow-tr-2009-1-flowvisor.pdf.
[100] A. Al-Shabibi, et al., Openvirtex: make your virtual SDNs programmable, in: Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, Chicago, Illinois, USA, 2014, pp. 25–30. doi: 10.1145/2620728.2620741.
[101] Z. Bozakov, P. Papadimitriou, Autoslice: automated and scalable slicing for software-defined networks, in: Proceedings of the 2012 ACM Conference on CoNEXT Student Workshop, Nice, France, 2012, pp. 3–4, doi:10.1145/2413247.2413251.
[102] H. Yamanaka, E. Kawai, S. Ishii, S. Shimojo, AutoVFlow: autonomous virtualization for wide-area openflow networks, in: Proceedings of the 2014 Third European Workshop on Software Defined Networks, London, UK, 2014, pp. 67–72, doi:10.1109/EWSDN.2014.28.
[103] Docker, [Online]. Available: https://www.docker.com.
[104] D. Drutskoy, E. Keller, J. Rexford, Scalable network virtualization in software-defined networks, IEEE Internet Comput. 17 (2) (2013) 20–27, doi:10.1109/MIC.2012.144.
[105] Libvirt virtualization API, [Online]. Available: https://libvirt.org.
[106] VMware, vcenter server, [Online]. Available: https://www.vmware.com/cn/products/vcenter-server.html.
[107] H. Gupta, S.B. Nath, S. Chakraborty, S.K. Ghosh, SDFog: A Software Defined Computing Architecture for QoS Aware Service Orchestration over Edge Devices, Cornell University Library, 2016, p. 1-8.
[108] J. Al-Badarneh, Y. Jararweh, M. Al-Ayyoub, M. Al-Smadi, R. Fontes, Software defined storage for cooperative mobile edge computing systems, in: Proceedings of the 2017 Fourth International Conference on Software Defined Systems (SDS), Valencia, Spain, 2017, pp. 174–179, doi:10.1109/SDS.2017.7939160.
[109] Ceph storage, [Online]. Available: http://ceph.com.
[110] P. Quinn, U. Elzur, Network service header, RFC 7665, 2017. [Online]. Available: https://tools.ietf.org/pdf/draft-ietf-sfc-nsh-13.pdf.
[111] Linux foundation, openswitch. [Online]. Available: http://www.openswitch.net/use/usehome.
[112] Openstack, distributed virtual router, [Online]. Available: https://wiki.openstack.org/wiki/Neutron/DVR.
[113] Brocade, vyatta 5600 vrouter, [Online]. Available: https:networkbuilders.intel.com/docs/Vyatta5600PerformanceTestFullReport.pdf.
[114] v. R. Ericsson, [Online]. Available: http://archive.ericsson.net/service/internet/picov/get?DocNo=1/28701-FGB1010557&Lang=EN&HighestFree=Y.
[115] Data plane development toolkit (DPDK), 2015, [Online]. Available: http://dpdk.org.
[116] P. Emmerich, D. Raumer, F. Wohlfart, G. Carle, Assessing soft- and hardware bottlenecks in PC-based packet forwarding systems, in: Proceedings of the Fourteenth International Conference on Networks, Barcelona, Spain, 2015, pp. 1–6.
[117] Linux foundation, FD.io - the fast data project, [Online]. Available: https://fd.io/.
[118] Linux foundation, IO visor project, [Online]. Available: https://www.iovisor.org/.
[119] Opendataplane project, [Online]. Available: https://www.opendataplane.org/.
[120] J.-H. Kim, J.-C. Na, A study on one-way communication using PF_RING ZC, in: Proceedings of the 19th International Conference on Advanced Communication Technology (ICACT), Bongpyeong, South Korea, 2017, pp. 301–304, doi:10.23919/ICACT.2017.7890102.
[121] D. Schönffmann, Comparing PFQ: a high-speed packet IO framework, Netw. Archit. Serv. (2016) 61–65, doi:10.2313/NET-2016-07-1_09.
[122] S. Han, K. Jang, K.S. Park, S. Moon, Packetshader: a GPU-accelerated software router, in: SIGCOMM, New Delhi, India, 2010, pp. 1–12. doi: 10.1145/1851275.1851207.
[123] L.-D. Chou, et al., The implementation of multilayer virtual network management system on netFPGA, in: Proceedings of the 2011 IEEE 17th International Conference on Parallel and Distributed Systems, Tainan, Taiwan, 2011, pp. 988–991, doi:10.1109/ICPADS.2011.135.
[124] TM forum, ZOOM, 2014, [Online]. Available: https://www.tmforum.org/wp-content/uploads/2014/10/ZoomDownload.pdf.
[125] D.R. Lopez, OpenMANO: the dataplane ready open source NFV MANO stack, in: IETF Meeting Proceedings, Dallas, Texas, USA, 2015.
[126] F.-J. R. Salguero, Open source MANO, 2016. [Online]. Available: https://osm.etsi.org/wikipub/images/5/5a/OSM_Introduction_Francisco.pdf.
[127] Openstack tacker. [Online]. Available: https://wiki.openstack.org/wiki/Tacker.
[128] Cloudify, [Online]. Available: http://cloudify.co.
[129] Logstash, [Online]. Available: https://github.com/elastic/logstash.
[130] RabbitMQ, [Online]. Available: https://github.com/rabbitmq.
[131] OPEN-o, [Online]. Available: https://www.open-o.org/.
[132] M. Bjorklund, YANG - a data modeling language for the network configuration protocol (NETCONF), RFC 6020, 2010.
[133] FROG v4.0, 2017, [Online]. Available: https://github.com/netgroup-polito/frog4.
[134] M.T. Beck, J.F. Botero, Scalable and coordinated allocation of service function chains, Elsevier Comput. Commun. 102 (1) (2017) 78–88, doi:10.1016/j.comcom.2016.09.010.
[135] B. Wang, M.-P. O., Lightweight VNF manager solution for virtual functions, J. ICT 3 (0) (2015) 91–104, doi:10.13052/jicts2245-800X.315.
[136] Linux foundation, opendaylight, [Online]. Available: https://www.opendaylight.org/.
[137] Infoblox, domain name system, 2016a, [Online]. Available: https://www.infoblox.com/wp-content/uploads/2016/04/infoblox-solution-virtual-secure-dns-secures-the-heart-of-service-provider-networks_0.pdf.
[138] NEC, NFV c-RAN for efficient RAN resource allocation, 2016b, [Online]. Available: http://www.nec.com/en/global/solutions/nsp/sc2/doc/wp_c-ran.pdf.
[139] NFWare, virtual carrier grade NAT, 2017, [Online]. Available: http://nfware.com/virtual-cgnat.
[140] Oracle, agile IMS network infrastructure for session delivery, 2013. [Online]. Available: http://www.oracle.com/us/industries/communications/agile-ims-network-ds-2062279.pdf.
[141] Red hat, enterprise linux atomic host, 2016, [Online]. Available: https://www.redhat.com/zh/resources/red-hat-enterprise-linux-atomic-host.
[142] 6wind, virtual accelerator, 2015, [Online]. Available: http://www.6wind.com/products/speedseries/6wind-virtual-accelerator/.
[143] 6wind, turbo IPsec, 2016, [Online]. Available: http://www.6wind.com/products/speedseries/6wind-turbo-ipsec/.
[144] Cisco, design and configuration guide: Best practices for virtual port channels (vPC) on cisco nexus 7000 series switches, 2016, [Online]. Available: http://www.cisco.com/c/dam/en/us/td/docs/switches/datacenter/sw/design/vpc_design/vpc_best_practices_design_guide.pdf.
[145] Cisco, virtual managed services platform, 2015. [Online]. Available: http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/virtual-managed-services/at-a-glance-c45-735822.pdf.
[146] Ericsson, virtual evolved packet core, 2016, [Online]. Available: https://www.ericsson.com/ourportfolio/it-and-cloud-products/virtual-evolved-packet-core.
[147] Juniper, vMX virtual router, 2017, [Online]. Available: https://www.juniper.net/assets/cn/zh/local/pdf/datasheets/1000522-en.pdf.
[148] Juniper, vSRX virtual firewall, 2017, [Online]. Available: http://www.juniper.net/us/en/local/pdf/datasheets/1000489-en.pdf.
[149] Nominum, n2 secure consumer, 2017, [Online]. Available: https://nominum.com/wp-content/uploads/2017/03/N2-Secure-Consumer-Solutions-Brief.pdf.
[150] Nominum, vantio cacheserve 7, 2017, [Online]. Available: https://nominum.com/wp-content/uploads/2015/06/Vantio-CacheServe7-DataSheet.pdf.
[151] F5, optimize and monetize networks with context-aware policy enforcement, 2016, [Online]. Available: https://www.f5.com/pdf/products/big-ip-policy-enforcement-manager-datasheet.pdf.
[152] F5, application delivery with programmable infrastructure, 2016, [Online]. Available: https://www.f5.com/pdf/products/big-ip-local-traffic-manager-ds.pdf.
[153] F5, deploy f5 application delivery and security services in private, public, and hybrid IT cloud environments, 2017, [Online]. Available: https://www.f5.com/pdf/products/big-ip-virtual-editions-datasheet.pdf.
[154] Metaswitch, perimeta session border controller, 2017, [Online]. Available: https://www.metaswitch.com/perimeta-session-border-controller-sbc.
[155] Metaswitch, clearwater project. [Online]. Available: http://www.projectclearwater.org/.
[156] Metaswitch, voice over LTE, [Online]. Available: https://www.sdxcentral.com/products/metaswitch-voice-over-lte/.
[157] Brocade, virtual mobile analytics, 2017, [Online]. Available: http://www.brocade.com/en/backend-content/pdf-page.html?/content/dam/common/documents/content-types/datasheet/brocade-analytics-monitoring-platform-ds.pdf.
[158] V.T.M. Brocade, 2016, [Online]. Available: https://www.brocade.com/content/dam/common/documents/content-types/datasheet/brocade-virtual-traffic-manager-ds.pdf.
[159] Brocade, virtual application delivery controller, 2016. [Online]. Available: https://www.brocade.com/content/dam/common/documents/content-types/datasheet/brocade-application-delivery-ds.pdf.
[160] J. Martins, et al., ClickOS and the art of network function virtualization, in: Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14), Seattle, WA, USA, 2014, pp. 459–473.
[161] A. Blenk, A. Basta, M. Reisslein, W. Kellerer, Survey on network virtualization hypervisors for software defined networking, IEEE Commun. Surv. Tutor. 18 (1) (2016) 655–685, doi:10.1109/COMST.2015.2489183.
[162] C.G. Kominos, N. Seyvet, K. Vandikas, Bare-metal, virtual machines and containers in openstack, in: Proceedings of the 2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), Paris, France, 2017, pp. 36–43. doi: 10.1109/ICIN.2017.7899247.
[163] Clear container, 2017, [Online]. Available: https://lwn.net/Articles/644675/.
[164] A. Madhavapeddy, D.J. Scott, Unikernels: the rise of the virtual library operating system, Commun. ACM 57 (1) (2014) 61–69, doi:10.1145/2541883.2541895.
[165] NFV report series, foundations of NFV: NFV infrastructure and VIM,

2017, [Online]. Available: https://www.sdxcentral.com/reports/2017/nfv-infrastructure-vim/.
[166] ETSI, NFV: Use cases, 2017, [Online]. Available: http://www.etsi.org/deliver/etsi_gr/NFV/001_099/001/01.02.01_60/gr_NFV001v010201p.pdf.
[167] R. Krishnan, N. Figueria, D. Krishnaswamy, D.R. Lopez, NFVIaas architectural framework for policy based resource placement and scheduling, IRTF, 2014. [Online]. Available: https://tools.ietf.org/html/draft-krishnan-nfvrg-policy-based-rm-nfviaas-00.
[168] E. Montariolo, Organization transformation for network function virtualization infrastructure as a service (NFVIaas), VMware, 2015, [Online]. Available: https://www.vmware.com/files/pdf/solutions/vmware-organization-transformation-nfv-iaas.pdf.
[169] A. Manzalini, N. Crespi, SDN and NFV for network cloud computing: a universal operating system for SD infrastructures, in: Proceedings of the IEEE Fourth Symposium on Network Cloud Computing and Applications (NCCA), Munich, 2015, pp. 1–6. doi: 10.1109/NCCA.2015.11.
[170] D. Cotroneo, et al., Dependability evaluation and benchmarking of network function virtualization infrastructures, in: Proceedings of the 1st IEEE Conference on Network Softwarization (NetSoft), London, 2015, pp. 1–9, doi:10.1109/NETSOFT.2015.7116123.
[171] R.V. Rosa, C.E. Rothenberg, R. Szabo, VBaas: VNF benchmark-as-a-service, in: Proceedings of the 2015 Fourth European Workshop on Software Defined Networks, Bilbao, 2015, pp. 79–84, doi:10.1109/EWSDN.2015.65.
[172] M. Zeng, W. Fang, Z. Zhu, Orchestrating tree-type VNF forwarding graphs in inter-DC elastic optical networks, J. Lightwave Technol. 34 (14) (2016) 3330–3341, doi:10.1109/JLT.2016.2565002.
[173] J.G. Herrera, J.F. Botero, Resource allocation in NFV: a comprehensive survey, IEEE Trans. Netw. Serv. Manag. PP (99) (2016), doi:10.1109/TNSM.2016.2598420. 1-1
[174] R. Mijumbi, et al., Server placement and assignment in virtualized radio access networks, in: Proceedings of the 11th International Conference on Network and Service Management (CNSM), Barcelona, 2015, pp. 398–401, doi:10.1109/CNSM.2015.7367390.
[175] M.Z. Shafiq, A.X. Liu, A.R. Khakpour, Revisiting caching in content delivery networks, in: Proceedings of the 2014 International Conference on Measurement and Modeling of Computer Systems, Austin, Texas, USA, 2014, pp. 567–568, doi:10.1145/2637364.2592021.
[176] M. Mangili, F. Martignon, A. Capone, Concise paper: stochastic planning for content delivery: unveiling the benefits of network functions virtualization, in: Proceedings of the 2014 IEEE 22nd International Conference on Network Protocols, Raleigh, NC, 2014, pp. 344–349, doi:10.1109/ICNP.2014.56.
[177] Huawei, NFV open lab, 2013, [Online]. Available: http://nfvopenlab.huawei.com/index.html.
[178] R. Riggio, T. Rasheed, F. Granelli, EmPOWER: a testbed for network function virtualization research and experimentation, in: IEEE SDN for Future Networks and Services (SDN4FNS), Trento, 2013, pp. 1–5, doi:10.1109/SDN4FNS.2013.6702538.
[179] K. Gomez, R. Riggio, T. Rasheed, D. Miorandi, F. Granelli, Energino: a hardware and software solution for energy consumption monitoring, in: Proceedings of the 10th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), Paderborn, Germany, 2012, pp. 311–317.
[180] R. Riggio, C. Sengul, L. Suresh, J. SchulzCzander, A. Feldmann, Thor: energy programmable wifi networks, in: Proceedings of the 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Turin, 2013, pp. 21–22, doi:10.1109/INFCOMW.2013.6970705.
[181] OpenSDNCore. [Online]. Available: http://www.opensdncore.org/.
[182] HP, openNFV, [Online]. Available: http://www8.hp.com/us/en/cloud/nfv-architecture.html.
[183] A. Boubendir, E. Bertin, N. Simoni, On-demand, dynamic and at-the-edge VNF deployment model application to web real-time communications, in: Proceedings of the 12th International Conference on Network and Service Management (CNSM), Montreal, QC, Canada, 2016, pp. 318–323, doi:10.1109/CNSM.2016.7818440.
[184] S. Sahhaf, et al., Network service chaining with optimized network function embedding supporting service decompositions, Elsevier Comput. Netw. 93 (3) (2015) 492–505, doi:10.1016/j.comnet.2015.09.035.
[185] S.Q. Zhang, et al., Routing algorithms for network function virtualization enabled multicast topology on SDN, IEEE Trans. Netw. Serv. Manag. 12 (4) (2015) 580–594, doi:10.1109/TNSM.2015.2465371.
[186] Y. Jin, Y. Wen, When cloud media meet network function virtualization: challenges and applications, IEEE MultiMedia 24 (3) (2017) 72–82, doi:10.1109/MMUL.2017.3051519.
[187] M.F. Bari, et al., Orchestrating virtualized network functions, IEEE Trans. Netw. Serv. Manag. PP (99) (2016) 1–14, doi:10.1109/TNSM.2016.2569020.
[188] R. Riggio, et al., Scheduling wireless virtual networks functions, IEEE Trans. Netw. Serv. Manag. 13 (2) (2016) 240–252, doi:10.1109/TNSM.2016.2549563.
[189] M.C. Luizelli, et al., Piecing together the NFV provisioning puzzle: efficient placement and chaining of virtual network functions, in: Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, 2015, pp. 98–106. doi: 10.1109/INM.2015.7140281.
[190] A. Gupta, M.F. Habib, P. Chowdhury, M. Tornatore, B. Mukherjee, Joint Virtual Network Function Placement and Routing of Traffic in Operator Network, University of California Davis, 2015, pp. 1–5.
[191] B. Addis, D. Belabed, M. Bouet, S. Secci, Virtual network functions placement and routing optimization, in: Proceedings of the IEEE 4th International Conference on Cloud Networking (CloudNet), Niagara Falls, ON, 2015, pp. 171–177, doi:10.1109/CloudNet.2015.7335301.
[192] IBM CPLEX optimizer. [Oline]. Available: https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer.
[193] LINGO, [Online]. Available: http://www.lindo.com/index.php/products/lingo-and-optimization-modeling.
[194] GNU linear programming kit (GLPK), 2012, [Online]. Available: https://www.gnu.org/software/glpk.
[195] H. Moens, F.D. Turck, VNF-p: a model for efficient placement of virtualized network functions, in: Proceedings of the 10th International Conference on Network and Service Management (CNSM) and Workshop, Rio de Janeiro, 2014, pp. 418–423, doi:10.1109/CNSM.2014.7014205.
[196] S. Khebbache, M. Hadji, D. Zeghlache, Scalable and cost-efficient algorithms for VNF chaining and placement problem, in: Proceedings of the 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), 2017, pp. 92–99, doi:10.1109/ICIN.2017.7899395.
[197] C. Pham, N.H. Tran, S. Ren, W. Saad, C.S. Hong, Traffic-aware and energy-efficient vNF placement for service chaining: joint sampling and matching approach, IEEE Trans. Serv. Comput. PP (99) (2017) 1–13, doi:10.1109/TSC.2017.2671867.
[198] F. Lucrezia, G. Marchetto, F. Risso, V. Vercellone, Introducing network-aware scheduling capabilities in openstack, in: Proceedings of the First IEEE International Conference on Network Softwarization (Netsoft 2015), London, UK, 2015, pp. 1–5, doi:10.1109/NETSOFT.2015.7116155.
[199] R. Mijumbi, et al., Design and evaluation of algorithms for mapping and scheduling of virtual network functions, in: Proceedings of the 1st IEEE Conference on Network Softwarization (NetSoft), London, 2015, pp. 1–9, doi:10.1109/NETSOFT.2015.7116120.
[200] L. Liu, Z. Qiu, A survey on virtual machine scheduling in cloud computing, in: Proceedings of the 2016 2nd IEEE International Conference on Computer and Communications (ICCC), Chengdu, China, 2016, pp. 2717–2721, doi:10.1109/CompComm.2016.7925192.
[201] J.F. Riera, E. Escalona, J. Batallé, E. Grasa, J.A. García-Espín, Virtual network function scheduling: Concept and challenges, in: Proceedings of the 2014 International Conference on Smart Communications in Network Technologies (SaCoNeT), Spain, 2014, pp. 1–5, doi:10.1109/SaCoNeT.2014.6867768.
[202] M. Shifrin, E. Biton, O. Gurewitz, Optimal control of VNF deployment and scheduling, in: Proceedings of the International Conference on the Science of Electrical Engineering (ICSEE), Eilat, Israel, 2016, pp. 1–5, doi:10.1109/ICSEE.2016.7806110.
[203] L. Qu, C. Assi, K. Shaban, Network function virtualization scheduling with transmission delay optimization, in: IEEE/IFIP Network Operations and Management Symposium (NOMS), Istanbul, 2016, pp. 638–644, doi:10.1109/NOMS.2016.7502870.
[204] L. Qu, C. Assi, K. Shaban, Delay-aware scheduling and resource optimization with network function virtualization, IEEE Trans. Commun. 64 (9) (2016) 3746–3758, doi:10.1109/TCOMM.2016.2580150.
[205] I.-K. Hatemet, et al., Network issues in virtual machine migration, in: Proceedings of the International Symposium on Networks, Computers and Communications (ISNCC), Hammamet, 2015, pp. 1–6, doi:10.1109/ISNCC.2015.7238579.
[206] L. Nobach, I. Rimac, V. Hilt, D. Hausheer, Slim: enabling efficient, seamless NFV state migration, in: Proceedings of the IEEE 24th International Conference on Network Protocols (ICNP), Singapore, 2016, pp. 1–2, doi:10.1109/ICNP.2016.7784459.
[207] V. Medina, J.M. Garcła, A survey of migration mechanisms of virtual machines, J. ACM Comput. Surv. (CSUR) 46 (3) (2014) 1–30, doi:10.1145/2492705.
[208] D. Kapil, E.S. Pilli, R.C. Joshi, Live virtual machine migration techniques: Survey and research challenges, in: Proceedings of the 3rd IEEE International Advance Computing Conference (IACC), Ghaziabad, 2013, pp. 963–969, doi:10.1109/IAdCC.2013.6514357.
[209] V. Eramo, M. Ammar, F.G. Lavacca, Migration energy aware reconfigurations of virtual network function instances in NFV architectures, IEEE Access 5 (0) (2017). 4927–2938, 10.1109/ACCESS.2017.2685437.
[210] V. Eramo, E. Miucci, M. Ammar, F.G. Lavacca, An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures, IEEE/ACM Trans. Netw. PP (99) (2017) 1–18, doi:10.1109/TNET.2017.2668470.
[211] G.D. Forney, The viterbi algorithm, in: IEEE Proceedings, 61, 1973, pp. 268–278, doi:10.1109/PROC.1973.9030.
[212] B. Andrus, A. Autenrieth, S. Pachnicke, J.J.V. Olmos, I.T. Monroy, Live migration downtime analysis of a VNF guest for a proposed optical FMC network architecture, IEEE Photonic Netw. (2016) 1–5.
[213] W. Cerroni, F. Callegati, Live migration of virtual network functions in cloud-based edge networks, in: Proceedings of the IEEE International Conference on Communications (ICC), Sydney, NSW, 2014, pp. 2963–2968, doi:10.1109/ICC.2014.6883775.
[214] I.-K. Hatem, et al., OMAC: optimal migration algorithm for virtual CDN, in: Proceedings of the 23rd International Conference on Telecommunications (ICT), Thessaloniki, 2016, pp. 1–6, doi:10.1109/ICT.2016.7500390.
[215] H. Wang, Y. Li, Y. Zhang, D. Jin, Virtual machine migration planning in software-defined networks, in: Proceedings of the IEEE Conference on Computer Communications (INFOCOM), Kowloon, 2015, pp. 487–495, doi:10.1109/INFOCOM.2015.7218415.
[216] M.F. Bari, et al., CQNCR: optimal VM migration planning in cloud data centers,

in: Proceedings of the IFIP Networking Conference, Trondheim, 2014, pp. 1–9, doi:10.1109/IFIPNetworking.2014.6857120.
[217] X. Li, Q. He, J. Chen, K. Ye, T. Yin, Informed live migration strategies of virtual machines for cluster load balancing, in: Proceedings of the International Conference on Network and Parallel Computing, Berlin, Heidelberg, 2011, pp. 111–122, doi:10.1007/978-3-642-24403-2_9.
[218] S. Rajagopalan, D. Williams, H. Jamjoom, A. Warfield, Split/merge: system support for elastic execution in virtual middleboxes, in: Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation, CA, 2013, pp. 227–240, doi:10.1145/1327452.1327492.
[219] G.J. Aaron, et al., OpenNF: enabling innovation in network function control, in: ACM SIGCOMM, Chicago, USA, 2014, pp. 163–174, doi:10.1145/2619239.2626313.
[220] L. Liu, H. Xu, Z. Niu, P. Wang, D. Han, U-HAUL: efficient state migration in NFV, in: Proceedings of the 7th ACM SIGOPS Asia-Pacific Workshop on Systems, Hong Kong, China, 2016, doi:10.1145/2967360.2967363.
[221] Y. Wang, G. Xie, Z. Li, P. He, K. Salamatian, Transparent flow migration for NFV, in: Proceedings of the IEEE 24th International Conference on Network Protocols (ICNP), Singapore, 2016, pp. 1–10, doi:10.1109/ICNP.2016.7784446.
[222] G.J. Aaron, A. Akella, Improving the safety, scalability, and efficiency of network function state transfers, in: Proceedings of ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization, London, UK, 2015, pp. 43–48, doi:10.1145/2785989.2785997.
[223] C. Bu, X. Wang, H. Cheng, M. Huang, K. Li, S.K. Das, Enabling adaptive routing service customization via the integration of SDN and NFV, J. Netw. Comput. Appl. 93 (1) (2017) 123–136, doi:10.1016/j.jnca.2017.05.010.
[224] G. Lee, M. Kim, S. Choo, S. Pack, Y. Kim, Optimal flow distribution in service function chaining, in: Proceedings of the 10th International Conference on Future Internet (CFI'15), Seoul, Republic of Korea, 2015, pp. 17–20, doi:10.1145/2775088.2775103.
[225] COIN-OR, 2011, [Online]. Available: https://projects.coin-or.org/Cbc.
[226] J. Liu, W. Lu, F. Zhou, P. Lu, Z. Zhu, On dynamic service function chain deployment and readjustment, IEEE Trans. Netw. Serv. Manag. (2017) 1–10, doi:10.1109/TNSM.2017.2711610.
[227] N. Bouten, R. Mijumbi, J. Serrat, J. Famaey, S. Latré, F.D. Turck, Semantically enhanced mapping algorithm for affinity-constrained service function chain requests, IEEE Trans. Netw. Serv. Manag. 14 (2) (2017) 317–330, doi:10.1109/TNSM.2017.2681025.
[228] A. Belbekkouche, M.M. Hasan, A. Karmouch, Resource discovery and allocation in network virtualization, IEEE Commun. Surv. Tutor. 14 (4) (2012) 1114–1128, doi:10.1109/SURV.2011.122811.00060.
[229] A. Fischer, et al., Virtual network embedding: a survey, IEEE Commun. Surv. Tutor. 15 (4) (2013) 1888–1906, doi:10.1109/SURV.2013.013013.00155.
[230] X. Zhang, Z. Huang, C. Wu, Z. Li, F.C.M. Lau, Online stochastic buy-sell mechanism for VNF chains in the NFV market, IEEE J. Select. Areas Commun. 35 (2) (2017) 392–405, doi:10.1109/JSAC.2017.2659259.
[231] J. Fan, Z. Ye, C. Guan, X. Gao, K. Ren, C. Qiao, GREP: guaranteeing reliability with enhanced protection in NFV, in: HotMiddlebox, London, United Kingdom, 2015, pp. 13–18, doi:10.1145/2785989.2786000.
[232] S. D'Oro, L. Galluccio, S. Palazzo, G. Schembra, Exploiting congestion games to achieve distributed service chaining in NFV networks, IEEE J. Select. Areas Commun. 35 (2) (2017) 407–420, doi:10.1109/JSAC.2017.2659298.
[233] B. Yi, X. Wang, M. Huang, Design and evaluation of schemes for provisioning service function chain with function scalability, J. Netw. Comput. Appl. 93 (1) (2017) 197–214, doi:10.1016/j.jnca.2017.05.013.
[234] T. Li, H. Zhou, H. Luo, A new method for providing network services: service function chain, Elsevier Opt. Switch. Netw. (2015) 1–9, doi:10.1016/j.osn.2015.09.005.
[235] M. Chen, et al., Markov approximation for combinatorial network optimization, IEEE Trans. Inf. Theory 59 (10) (2013) 6301–6327, doi:10.1109/TIT.2013.2268923.
[236] Y. Nam, S. Song, J.M. Chung, Clustered NFV service chaining optimization in mobile edge clouds, IEEE Commun. Lett. 21 (2) (2017) 350–353, doi:10.1109/LCOMM.2016.2618788.
[237] P. Bellavista, et al., Virtual network function embedding in real cloud environments, Elsevier Comput. Netw. (2015) 506–517, doi:10.1016/j.comnet.2015.09.034.
[238] E. Kavvadia, S. Sagiadinos, K. Oikonomou, G. Tsioutsiouliklisa, S. Aissa, Elastic virtual machine placement in cloud computing network environments, Elsevier Comput. Netw. (2015) 435–447, doi:10.1016/j.comnet.2015.09.038.
[239] S. Kumar, M. Tufail, S. Majee, C. Captari, S. Homma, Service function chaining use cases in data centers, IETF (2017). [Online]. Available: https://tools.ietf.org/html/draft-ietf-sfc-dc-use-cases-06.
[240] H. Zhang, et al., Service chain header, IETF Network working group (2014) 1–14. [Online]. Available: https://tools.ietf.org/pdf/draft-zhang-sfc-sch-00.pdf.
[241] H. Song, et al., SFC header mapping for legacy SF, IETF SFC working group (2017) 1–11. [Online]. Available: https://tools.ietf.org/pdf/draft-song-sfc-legacy-sf-mapping-08.pdf.
[242] S. Previdi, et al., IPv6 segment routing header (SRH), IETF Network working group (2017) 1–32. [Online]. Available: https://tools.ietf.org/pdf/draft-ietf-6man-segment-routing-header-07.pdf
[243] W. Ding, W. Qi, J. Wang, B. Chen, OpenSCaas: an open service chain as a service platform toward the integration of SDN and NFV, IEEE Netw. 29 (3) (2015) 30–35, doi:10.1109/MNET.2015.7113222.
[244] S.K. Fayazbakhsh, et al., Flowtags: Enforcing network-wide policies in the presence of dynamic middlebox actions, in: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, Hong Kong, China, 2013, pp. 19–24, doi:10.1145/2491185.2491203.
[245] S. Misra, S. Goswami, Routing and MPLS Traffic Engineering, Network Routing: Fundamentals, Applications, and Emerging Technologies, Wiley Telecom, 2014, doi:10.1002/9781119114864.ch5.
[246] Y. Sinha, S. Bhatia, V.S. Shekhawat, G.S.S. Chalapathi, MPLS based hybridization in SDN, in: Proceedings of the 2017 Fourth International Conference on Software Defined Systems (SDS), Valencia, Spain, 2017, pp. 156–161, doi:10.1109/SDS.2017.7939157.
[247] X. Li, M.J. Freedman, Scaling IP Multicast on Datacenter Topologies, Santa Barbara, California, USA, 2013, pp. 61–72, doi:10.1145/2535372.2535380.
[248] S.Q. Zhang, et al., Joint NFV placement and routing for multicast service on SDN, in: Proceedings of the IEEE/IFIP Network Operations and Management Symposium, Istanbul, 2016, pp. 333–341, doi:10.1109/NOMS.2016.7502829.
[249] S. Vrontis, S. Xynogalas, E. Sykas, Steiner tree compilation of multicast under differentiated services constraints, J. Commun. Netw. 9 (1) (2007) 84–92, doi:10.1109/JCN.2007.6182817.
[250] J. Byrka, F. Grandoni, T. RothvoB, L. Sanit, An improved LP-based approximation for steiner tree, in: Proceedings of the Forty-Second ACM Symposium on Theory of Computing, Cambridge, Massachusetts, USA, 2010. 5863–592, doi:10.1137/S0097539792236237.
[251] X. Gao, Z. Ye, W. Zhong, C. Qiao, X. Cao, H. Zhao, H. Yu, V. Anand, Multicast service-oriented virtual network mapping over elastic optical networks, in: Proceedings of the IEEE International Conference on Communications (ICC), London, 2015, pp. 5174–5179, doi:10.1109/ICC.2015.7249145.
[252] X. Gao, et al., Virtual network mapping for reliable multicast services with max-min fairness, in: Proceedings of the IEEE Global Communications Conference (GLOBECOM), San Diego, CA, 2015, pp. 1–6, doi:10.1109/GLOCOM.2015.7417549.
[253] M. Zeng, et al., Orchestrating multicast-oriented NFV trees in inter-DC elastic optical networks, in: Proceedings of the IEEE International Conference on Communications (ICC), Kuala Lumpur, 2016, pp. 1–6, doi:10.1109/ICC.2016.7511588.
[254] B. Yi, X. Wang, M. Huang, A. Dong, A multi-stage solution for NFV-enabled multicast over the hybrid infrastructure, IEEE Commun. Lett. 21 (9) (2017) 2061–2064, doi:10.1109/LCOMM.2017.2715808.
[255] S.Q. Zhang, et al., Network function virtualization enabled multicast routing on SDN, in: Proceedings of the IEEE International Conference on Communications (ICC), London, 2015, pp. 5595–5601, doi:10.1109/ICC.2015.7249214.
[256] D. Kreutz, P.E. Verissimo, S. Azodolmolky, Software-defined networking: a comprehensive survey, Proc. IEEE 103 (1) (2015) 14–76, doi:10.1109/JPROC.2014.2371999.
[257] Y. Zhao, H. Wang, X. Lin, T. Yu, C. Qian, Pronto: Efficient test packet generation for dynamic network data planes, in: Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 2017, pp. 13–22, doi:10.1109/ICDCS.2017.55.
[258] F. Pianese, M. Gallo, A. Conte, D. Perino, Orchestrating 5g virtual network functions as a modular programmable data plane, in: Proceedings of the 2016 IEEE/IFIP Network Operations and Management Symposium (NOMS 2016), Istanbul, Turkey, 2016, pp. 1305–1308, doi:10.1109/NOMS.2016.7503008.
[259] C. Sun, J. Bi, Z. Zheng, H. Hu, SLA-NFV: an SLA-sware high performance framework for NFV, in: 2016 SIGCOMM, Florianopolis, Brazil, 2016, pp. 581–582, doi:10.1145/2934872.2959058.
[260] R. Verret, S. Thompson, Custom FPGA-based tests with COTS hardware and graphical programming, in: 2010 IEEE AUTOTESTCON, Orlando, FL, USA, 2010, pp. 1–5, doi:10.1109/AUTEST.2010.5613619.
[261] B. Ferencz, T. Kovácsházy, Hardware assisted COTS IEEE 1588 solution for x86 linux and its performance evaluation, in: Proceedings of the 2013 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), Lemgo, Germany, 2013, pp. 47–52, doi:10.1109/ISPCS.2013.6644762.
[262] P. Velmyl, Developing application specific integrated circuits(ASIC) for cloud computing, in: Proceedings of the 2011 International Conference on Electronics, Communication and Computing Technologies, Pauls Nagar, India, 2011, pp. 43–46, doi:10.1109/ICECCT.2011.6077067.
[263] X. Fei, F. Liu, H. Xu, H. Jin, Towards load-balanced VNF assignment in geo-distributed NFV infrastructure, in: Proceedings of the 2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS), Vilanova i la Geltru, Spain, 2017, pp. 1–10, doi:10.1109/IWQoS.2017.7969166.
[264] Y. Wang, M. Wen, C. Zhang, J. Lin, RVNet: a fast and high energy efficiency network packet processing system on RISC-v, in: Proceedings of the 2017 IEEE 28th International Conference on Application-specific Systems, Architectures and Processors (ASAP), Seattle, WA, USA, USA, 2017, doi:10.1109/ASAP.2017.7995266.
[265] Z. Bronstein, E. Roch, J. Xia, A. Molkho, Uniform handling and abstraction of NFV hardware accelerators, IEEE Netw. 29 (3) (2015) 22–29, doi:10.1109/MNET.2015.7113221.
[266] S. Natarajan, et al., An analysis of lightweight virtualization technologies for NFV, IETF NFV Research Group (2016) 1–16. [Online]. Available: https://tools.ietf.org/html/draft-natarajan-nfvrg-containers-for-nfv-02.
[267] R. Shi, et al., MDP and machine learning-based cost-optimization of dynamic resource allocation for network function virtualization, in: Proceedings of the IEEE International Conference on Services Computing, New York, NY, 2015, pp. 65–73, doi:10.1109/SCC.2015.19.
[268] Y. Nakagawa, et al., Dynamic virtual network configuration between containers using physical switch functions for NFV infrastructure, in: Proceedings

of the 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), San Francisco, CA, USA, 2015, pp. 156–162, doi:10.1109/NFV-SDN.2015.7387421.
[269] Y. Zhang, et al., A communication-aware container re-distribution approach for high performance VNFs, in: Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 2017, pp. 1555–1564, doi:10.1109/ICDCS.2017.10.
[270] Openstack API, 2017, [Online]. Available: https://developer.openstack.org/api-guide/quick-start/.
[271] Nokia, NFV - a devops methodology for deployment, 2016. [Online]. Available: https://resources.ext.nokia.com/asset/200330.
[272] Openstack, tap as a service, 2017, [Online]. Available: https://github.com/openstack/tap-as-a-service.
[273] E. Maini, A. Manzalini, Management and orchestration of virtualized network functions, in: Monitoring and Securing Virtualized Networks and Services, Lecture Notes in Computer Science, volume 8508(0), Springer Berlin Heidelberg, 2014, pp. 52–56, doi:10.1007/978-3-662-43862-6_6.
[274] Cisco, elastic service controller, 2015, [Online]. Available: https://www.cisco.com/c/en/us/products/cloud-systems-management/elastic-services-controller-esc/index.html.
[275] Juniper, managing the VNF life cycle, 2016, [Online]. Available: https://www.juniper.net/documentation/en_US/junos/topics/task/configuration/vnf-lifecycle-management.html.
[276] S. Beereddy, K. Sirupa, NFV use case ??? delivering virtual CPE with multi-vendor VNF orchestration, in: Proceedings of the 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), San Francisco, CA, USA, 2015, pp. 25–27. doi: 10.1109/NFV-SDN.2015.7387399.
[277] IETF, Network configuration protocol (NETCONF), RFC 6241, 2011, [Online]. Available: https://tools.ietf.org/pdf/rfc6241.pdf.
[278] R. Riggio, et al., Virtual network functions orchestration in wireless networks, in: Proceedings of the 2015 11th International Conference on Network and Service Management (CNSM), Barcelona, 2015, pp. 108–116, doi:10.1109/CNSM.2015.7367346.
[279] H.M. Sabi, F.-M.E. Uzoka, K. Langmia, F.N. Njeh, Conceptualizing a model for adoption of cloud computing in education, Elsevier Int. J. Inf. Manag. 36 (2) (2015) 183–191, doi:10.1016/j.ijinfomgt.2015.11.010.
[280] A. Baumgartner, V.S. Reddy, T. Bauschert, Mobile core network virtualization: a model for combined virtual core network function placement and topology optimization, in: Proceedings of the 1st IEEE Conference on Network Softwarization (NetSoft), London, 2015, pp. 1–9, doi:10.1109/NETSOFT.2015.7116162.
[281] C. Pham, N.H. Tran, S. Ren, W. Saad, C.S. Hong, Traffic-aware and energy efficient vNF placement for service chaining: joint sampling and matching approach, IEEE Trans. Serv. Comput. PP (99) (2017) 1–14, doi:10.1109/TSC.2017.2671867.
[282] E.J. Scheid, et al., Policy-based dynamic service chaining in network functions virtualization, in: Proceedings of the IEEE Symposium on Computers and Communication (ISCC), Messina, 2016, pp. 340–345, doi:10.1109/ISCC.2016.7543763.
[283] M. Mechtri, C. Ghribi, D. Zeghlache, A scalable algorithm for the placement of service function chains, IEEE Trans. Netw. Serv. Manag. 13 (3) (2016) 533–546, doi:10.1109/TNSM.2016.2598068.
[284] X.W. Wang, et al., A flexible and generalized framework for access network selection in heterogeneous wireless networks, Pervasive Mobile Comput. (2017) 1–21, doi:10.1016/j.pmcj.2017.01.001.
[285] Intel, policy enforcement in SDN and NFV-based telco infrastructure, 2013. [Online]. Available: http://www.qosmos.com/wp-content/uploads/2013/09/Intel-Qosmos_AppNotes_Policy-Enforcement-in-SDN-NFV.pdf.
[286] M. Dobrescu, K. Argyraki, S. Ratnasamy, Toward predictable performance in software packet-processing platforms, 2012. 1–14. doi:10.1145/63404.63407.
[287] M. Skulysh, O. Klimovych, Approach to virtualization of evolved packet core network functions, in: Proceedings of the 2015 13th International Conference on Experience of Designing and Application of CAD Systems in Microelectronics (CADSM), Lviv, 2015, pp. 193–195, doi:10.1109/CADSM.2015.7230833.
[288] S. Gallenmller, P. Emmerich, F. Wohlfart, D. Raumer, G. Carle, Comparison of framework for high-performance packet i/o, in: Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for networking and communications systems, 2015, pp. 29–38, doi:10.1145/1851275.1851207.
[289] J.G. Luis, et al., High-performance network traffic processing systems using commodity hardware, in: Data Traffic Monitoring and Analysis, 7754, Springer Verlag Berlin Heidelberg, 2013, pp. 3–27, doi:10.1007/978-3-642-36784-7_1.
[290] J.J.G. Herrera, J.F.B. Vega, Network functions virtualization: a survey, IEEE Latin Am. Trans. 14 (2) (2016) 983–997, doi:10.1109/TLA.2016.7437249.
[291] Y. Nakajima, H. Masutani, H. Takahashi, High-performance vNIC framework for hypervisor-based NFV with userspace vswitch, Proceedings of the IEEE 4th European Workshop on Software Defined Networks, Bilbao (2015) 43–48. doi:10.1109/EWSDN.2015.59.
[292] L. Li, Y. Shi, J. Wang, Z. H., A VM-friendly NIC architecture for cloud computing, in: Proceedings of the 2017 IEEE 2nd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), Chengdu, China, 2017, pp. 196–200, doi:10.1109/ICCCBDA.2017.7951910.
[293] A. Morton, Considerations for benchmarking for virtual network functions and their infrastructure, 2015. 1–11. doi:10.17487/RFC2330.
[294] S. Lange, et al., Performance benchmarking of a software-based LTE SGW, in: Proceedings of the 2015 11th International Conference on Network and Service Management (CNSM), Barcelona, 2015, pp. 378–383, doi:10.1109/CNSM.2015.7367386.
[295] M. Chiosi, D. Clarke, P.W. Cablelabs, S. Wright, Network functions virtualisation: network operator perspectives on industry progress, in: SDN and OpenFlow World Congress, Dusseldorf-Germany, 3, 2014, pp. 1–16, doi:10.13140/RG.2.1.4110.2883.
[296] J. Keeney, S.V. Meer, L. Fallon, Towards real-time management of virtualized telecommunication networks, in: Proceedings of the 10th International Conference on Network and Service Management (CNSM) and Workshop, Rio de Janeiro, 2015, pp. 388–393, doi:10.1109/CNSM.2014.7014200.
[297] A. Hmaity, M. Savi, F. Musumeci, M. Tornatore, A. Pattavina, Virtual network function placement for resilient service chain provisioning, in: Proceedings of the 8th International Workshop on Resilient Networks Design and Modeling (RNDM), Halmstad, 2016, pp. 245–252, doi:10.1109/RNDM.2016.7608294.
[298] K. Suksomboon, M. Fukushima, S. Okamoto, M. Hayashi, A dilated-CPU-consumption-based performance prediction for multi-core software routers, in: Proceedings of the IEEE NetSoft Conference and Workshops (NetSoft), Seoul, 2016, pp. 193–201, doi:10.1109/NETSOFT.2016.7502413.
[299] T. Meyer, F. Wohlfart, D. Raumer, B.E. Wolfinger, G. Carle, Validated model-based performance prediction of multi-core software routers, Praxis der Informationsverarbeitung und Kommunikation (PIK) 37 (2) (2014), doi:10.1515/pik-2014-0006.
[300] Network simulator 3 (NS3), [Online]. Available: https://www.nsnam.org/.
[301] C. Xu, X. Chen, R.P. Dick, Z.M. Mao, Cache contention and application performance prediction for multi-core systems, in: Proceedings of the 2010 IEEE International Symposium on Performance Analysis of Systems & Software (ISPASS), White Plains, NY, 2010, pp. 76–86, doi:10.1109/ISPASS.2010.5452065.
[302] A. Lombardo, A. Manzalini, V. Riccobene, G. Schembra, An analytical tool for performance evaluation of software defined networking services, in: Proceedings of the 2014 IEEE Network Operations and Management Symposium (NOMS), Krakow, 2014, pp. 1–7, doi:10.1109/NOMS.2014.6838416.
[303] S. Fugkeaw, H. Sato, Enabling constraints and dynamic preventive access control policy enforcement in the cloud, in: Proceedings of the 10th International Conference on Availability, Reliability and Security (ARES), Toulouse, 2015, pp. 576–583, doi:10.1109/ARES.2015.33.
[304] X. Wang, W. Shi, Y. Xiang, J. Li, Efficient network security policy enforcement with policy space analysis, IEEE/ACM Trans. Netw. PP (99) (2015) 1–13, doi:10.1109/TNET.2015.2502402.
[305] L. Ma, W. Li, X. Qiu, Policy based traffic offload management mechanism in h(e)NB subsystem, in: Proceedings of the 13th Asia-Pacific Network Operations and Management Symposium, Taipei, Taiwan, 2011, pp. 1–6, doi:10.1109/APNOMS.2011.6077013.
[306] A.J. Mayer, X. Jiang, S. McCarron, H.-P. Chang, Policy exchange information model in support of cross-domain management, in: Proceedings of the 2010 International Conference on Network and Service Management, Canada, 2010, pp. 222–225, doi:10.1109/CNSM.2010.5691299.
[307] D. Vengerov, Dynamic adaptation of user migration policies in distributed virtual environments, in: Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 2017, pp. 4028–4035, doi:10.1109/IJCNN.2017.7966364.
[308] N. Figueira, R.R. Krishnan, SDN multi-domain orchestration and control: challenges and innovative future directions, in: Proceedings of the International Conference on Computing, Networking and Communications (ICNC), Garden Grove, CA, 2015, pp. 406–412, doi:10.1109/ICCNC.2015.7069378.
[309] Floodlight, [Online]. Available: http://www.projectfloodlight.org/.
[310] X. Li, C. Qian, An NFV orchestration framework for interference-free policy enforcement, in: Proceedings of the IEEE 36th International Conference on Distributed Computing Systems (ICDCS), Nara, 2016, pp. 649–658, doi:10.1109/ICDCS.2016.24.
[311] B.-I. Yaniv, et al., EnforSDN: network policies enforcement with SDN, in: Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, 2015, pp. 80–88, doi:10.1109/INM.2015.7140279.
[312] Z. Wang, N.H. Minsky, Establishing global policies over decentralized online social networks, in: Proceedings of the 10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, Miami, FL, USA, 2014, pp. 511–518.
[313] C. Araguz, On autonomous software architectures for distributed spacecraft: a local-global policy, in: Proceedings of the 2015 IEEE Aerospace Conference, Big Sky, MT, USA, 2015, pp. 1–9, doi:10.1109/AERO.2015.7119182.
[314] D. Li, et al., AHI: Efficient policy space set operations, in: Proceedings of the IEEE ICC 2017 SAC Symposium SDN & NFV Track, Paris, France, 2017, doi:10.1109/ICC.2017.7996842.
[315] R. Krishnan, T. Hinrichs, D. Krishnawamy, R. Krishnaswamy, Policy-based monitoring and energy management for NFV data centers, in: Proceedings of the International Conference on Computing and Network Communications (CoCoNet 2015), Trivandrum, India, 2015, pp. 10–17, doi:10.1109/CoCoNet.2015.7411161.
[316] M. Ayache, M. Erradi, B. Freisleben, Access control policies enforcement in a cloud environment: openstack, in: Proceedings of the 2015 11th International Conference on Information Assurance and Security (IAS), Marrakech, Morocco, 2016, pp. 26–31, doi:10.1109/ISIAS.2015.7492740.
[317] C. Lorenz, et al., An SDN/NFV-enabled enterprise network architecture offering fine-grained security policy enforcement, IEEE Commun. Mag. 55 (3) (2017) 217–223, doi:10.1109/MCOM.2017.1600414CM.
[318] A. Shaghaghi, et al., Towards policy enforcement point as a service (PEPS), in:

Proceedings of the 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Palo Alto, CA, USA, 2016, pp. 50–55, doi:10.1109/NFV-SDN.2016.7919475.
[319] S.K. Fayazbakhsh, et al., Flowtags: Enforcing network-wide policies in the presence of dynamic middlebox actions, in: Proceedings of the second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, Hong Kong, China, 2013, pp. 19–24, doi:10.1145/2491185.2491203.
[320] Q. He, Y. Wang, W. Li, X. Qiu, Traffic steering of middlebox policy chain based on SDN, in: Proceedings of the 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Lisbon, Portugal, Portugal, 2017, pp. 754–759, doi:10.23919/INM.2017.7987371.
[321] C. Basile, et al., A novel approach for integrating security policy enforcement with dynamic network virtualization, in: Proceedings of the 1st IEEE Conference on Network Softwarization (NetSoft), London, 2015, pp. 1–5, doi:10.1109/NETSOFT.2015.7116152.
[322] China telecom corporation annual report, 2012, [Online]. Available: http://www.chinatelecom-h.com/en/ir/report/annual2012.pdf.
[323] Greenpeace, clicking clean: How companies are creating the green internet, 2014. [Online]. Available: www.greenpeace.org.
[324] Bell labs, g.w.a.t.t. white paper, 2015, [Online]. Available: http://gwatt.net/intro/1.
[325] R. Mijumbi, J. Serrat, J.-L. Gorricho, J. Rubio-Loyola, On the Energy Efficiency Prospects of Network Function Virtualization, Springer Computer Science, 2015.
[326] Z. Xu, F. Liu, T. Wang, H. Xu, Demystifying the energy efficiency of NFV, IEEE/ACM 24th International Symposium on Quality of Service (IWQoS), 2016, doi:10.1109/IWQoS.2016.7590429.
[327] G. Tang, W. Jiang, Z. Xu, F. Liu, K. Wu, Zero-cost, fine-grained power monitoring of datacenters using nonintrusive power disaggregation, in: 2015 Middleware Conference, Vancouver, Canada, 2015, pp. 271–282, doi:10.1145/2814576.2814728.
[328] R. Bolla, C. Lombardo, R. Bruschi, S. Mangialardi, DROPv2: energy efficiency through network function virtualization, IEEE Netw. 28 (2) (2014) 26–32, doi:10.1109/MNET.2014.6786610.
[329] S. Luo, et al., Improving energy efficiency in industrial wireless sensor networks using SDN and NFV, in: Proceedings of the IEEE 83rd Vehicular Technology Conference (VTC Spring), Nanjing, 2016, pp. 1–5, doi:10.1109/VTCSpring.2016.7504281.
[330] T. Kim, T. Koo, E. Paik, SDN and NFV benchmarking for performance and reliability, in: Proceedings of the 17th Asia-Pacific Network Operations and Management Symposium (APNOMS), Busan, 2015, pp. 600–603, doi:10.1109/APNOMS.2015.7275403.
[331] J. Liu, et al., Reliability evaluation for NFV deployment of future mobile broadband networks, IEEE Wirel. Commun. 23 (3) (2016) 90–96, doi:10.1109/MWC.2016.7498079.
[332] X. Gao, Virtual network mapping for multicast services with maxCMin fairness of reliability, IEEE/OSA J. Opt. Commun. Netw. 7 (9) (2015) 942–951, doi:10.1364/JOCN.7.000942.
[333] D.V. Bernardo, B.B. Chua, Introduction and analysis of SDN and NFV security architecture (SN-SECA), in: Proceedings of the 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, Gwangiu, 2015, pp. 796–801, doi:10.1109/AINA.2015.270.
[334] B. Jaeger, Security orchestrator introducing a security orchestrator in the context of the ETSI NFV reference architecture, in: Proceedings of the 2015 IEEE Trustcom/BigDataSE/ISPA, Helsinki, 2015, pp. 1255–1260, doi:10.1109/Trustcom.2015.514.
[335] Telco systems, protecting your SDN and NFV network from cyber security vulnerabilities with full perimeter defense, 2015. [Online]. Available: http://www.telco.com/blog/.
[336] W. Yang, C. Fung, A survey on security in network functions virtualization, in: IEEE NetSoft Conference and Workshops (NetSoft), Seoul, 2016, pp. 15–19, doi:10.1109/NETSOFT.2016.7502434.
[337] M. Liyanage, et al., Leveraging LTE security with SDN and NFV, in: Proceedings of the IEEE 10th International Conference on Industrial and Information Systems (ICIIS), Peradeniya, 2015, pp. 220–225, doi:10.1109/ICIINFS.2015.7399014.
[338] T. Park, et al., QoSE: Quality of security a network security framework with distributed NFV, in: Proceedings of the IEEE International Conference on Communications (ICC), Kuala Lumpur, 2016, pp. 1–6, doi:10.1109/ICC.2016.7510777.
[339] Brocade, real-time SDN and NFV analytics and DDos mitigation, 2014. [Online]. Available: https://www.brocade.com/content/dam/common/documents/content-types/datasheet/brocade-interop-sdn-ddos-mitigation-flyer.pdf.
[340] X. Xu, C. Pautasso, L. Zhu, The blockchain as a software connector, in: Proceedings of the 13th Working IEEE/IFIP Conference on Software Architecture (WICSA), Venice, 2016, pp. 182–191, doi:10.1109/WICSA.2016.21.
[341] ARBOR networks, worldwide infrastructure security report, 2016. Pp. 1-26, [Online]. Available: https://pages.arbornetworks.com/rs/arbor/images/WISR2014.pdf.
[342] Huawei white paper, observation to NFV, 2014,
[343] Wedge networks white paper, network functions virtualization for security (NFV-s), 2014b, [Online]. Available: http://www.wedgenetworks.com/lit/Wedge%20Whitepaper%20NFV-S-06032014.pdf.
[344] L.R. Battula, Network security function virtualization(NSFV) towards cloud computing with NFV over openflow infrastructure: challenges and novel approaches, in: Proceedings of the 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), New Delhi, 2014, pp. 1622–1628, doi:10.1109/ICACCI.2014.6968453.
[345] W. Xia, et al., A survey on software-defined networking, IEEE Commun. Surv. Tutor. 17 (1) (2015) 27–51, doi:10.1109/COMST.2014.2330903.
[346] D.A. Temesgene, J.N. Martłnez, P. Dini, Softwarization and optimization for sustainable future mobile networks: a survey, IEEE Access PP (90) (2017). 1–1, 10.1109/ACCESS.2017.2771938.
[347] 5GMF white paper, 5g mobile communications systems for 2020 and beyond, 2017. [Online]. Available: http://5gmf.jp/wp/wp-content/uploads/2016/08/5GMF_WP100_Summary-E.pdf.
[348] L. Mamatas, S. Clayman, A. Galis, A service-aware virtualized software-defined infrastructure, IEEE Commun. Mag. 53 (4) (2015) 166–174, doi:10.1109/MCOM.2015.7081091.
[349] M.A. McCarthy, L.M. Herger, S.M. Khan, A compliance aware software defined infrastructure, in: Proceedings of the 2014 IEEE International Conference on Services Computing, Anchorage, AK, USA, 2014, pp. 560–567, doi:10.1109/SCC.2014.79.
[350] I. Bolodurina, D. Parfenov, Model control of traffic by using data flows classification of the cloud applications in software-defined infrastructure of virtual data center, in: Proceedings of the 2017 40th International Conference on Telecommunications and Signal Processing (TSP), Barcelona, Spain, 2017, pp. 8–11, doi:10.1109/TSP.2017.8075926.
[351] U. Zakia, H.B. Yedder, Dynamic load balancing in SDN-based data center networks, in: Proceedings of the 2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Vancouver, BC, Canada, 2017, pp. 242–247, doi:10.1109/IEMCON.2017.8117206.
[352] J. Lin, et al., Monarch: Monitoring and analytics in software defined infrastructures, in: Proceedings of the 2015 IEEE 4th International Conference on Cloud Networking (CloudNet), Niagara Falls, ON, Canada, 2015, pp. 198–200, doi:10.1109/CloudNet.2015.7335305.
[353] T. Abar, A.B. Letaifa, S.E. Asmi, Machine learning based qoe prediction in SDN networks, in: Proceedings of the 2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC), Valencia, Spain, 2017, pp. 1395–1400, doi:10.1109/IWCMC.2017.7986488.
[354] A. Montazerolghaem, M.H. Yaghmaee, A.L. Garcia, OpenAMI: software-defined AMI load balancing, IEEE Internet Things J. PP (99) (2017), doi:10.1109/JIOT.2017.2778006. 1–1
[355] P. Yasrebi, S. Monfared, H. Bannazadeh, A.L. Garcia, Security function virtualization in software defined infrastructure, in: Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, Canada, 2015, pp. 778–781, doi:10.1109/INM.2015.7140374.
[356] J. Duan, Y. Yang, A data center virtualization framework towards load balancing and multi-tenancy, in: Proceedings of the 2016 IEEE 17th International Conference on High Performance Switching and Routing (HPSR), Yokohama, Japan, 2016, pp. 14–21, doi:10.1109/HPSR.2016.7525633.
[357] G. Ghoda, A survey on data center network virtualization, in: Proceedings of the 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2016, pp. 3464–3470.
[358] J. Son, A.V. Dastjerdi, R.N. Calheiros, R. Buyya, SLA-aware and energy-efficient dynamic overbooking in SDN-based cloud data centers, IEEE Trans. Sustain. Comput. 2 (2) (2017) 76–89, doi:10.1109/TSUSC.2017.2702164.
[359] Y. Sun, X. Hu, X. Liu, X. He, K. Wang, A software-defined green framework for hybrid EV-charging networks, IEEE Commun. Mag. 55 (11) (2017) 62–69, doi:10.1109/MCOM.2017.1601190.
[360] HP enterprise, composable infrastructure, 2017. [Online]. Available: https://www.hpe.com/us/en/solutions/infrastructure/composable-infrastructure.html.
[361] M. Huang, W. Liang, Incremental SDN-enabled switch deployment for hybrid software-defined networks, in: Proceedings of the 2017 26th International Conference on Computer Communication and Networks (ICCCN), Vancouver, BC, Canada, 2017, pp. 1–6, doi:10.1109/ICCCN.2017.8038498.
[362] T. Wang, F. Liu, H. Xu, An efficient online algorithm for dynamic SDN controller assignment in data center networks, IEEE/ACM Trans. Netw. 25 (5) (2017) 2788–2801, doi:10.1109/TNET.2017.2711641.
[363] S.V. Galich, M.S. Deogenov, E.S. Semenov, Control traffic parameters analysis in various software-defined networking topologies, in: Proceedings of the 2017 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), St. Petersburg, Russia, 2017, pp. 1–6, doi:10.1109/ICIEAM.2017.8076439.
[364] F. Pakzad, M. Portmanna, W.L. Tanb, J. Indulskaa, Efficient topology discovery in openflow-based software defined networks, Comput. Commun. 77 (0) (2016) 52–61, doi:10.1016/j.comcom.2015.09.013.
[365] R. Wen, et al., Protocol function block mapping of software defined protocol for 5g mobile networks, IEEE Trans. Mobile Comput. PP (99) (2017). 1–1, doi:10.1109/TMC.2017.2687919.
[366] R. Abeysekara, et al., Three-tier private cloud application management system with on-site hardware, in: Proceedings of the 2017 16th International Symposium on Parallel and Distributed Computing (ISPDC), Innsbruck, Austria, 2017, pp. 54–61, doi:10.1109/ISPDC.2017.22.
[367] K. Li, C. Wang, Q. Shen, J. Wang, Q. Qi, MVNC: a SDN-based multi-tenant virtual network customization mechanism in cloud data center, in: Proceedings of the 2016 International Conference on Networking and Network Applications (NaNA), Hakodate, Japan, 2016, pp. 239–243, doi:10.1109/NaNA.2016.51.
[368] Y. Wang, Y. Zhang, J. Chen, Pursuing differentiated services in a SDN-based iot-oriented pub/sub system, in: Proceedings of the 2017 IEEE International Conference on Web Services (ICWS), Honolulu, HI, USA, 2017. 906–906, doi:10.1109/ICWS.2017.118.

[369] S. Ahmed, M. Raja, Software defined networks for multitenant, multiplatform applications, in: Proceedings of the 2017 14th International Conference on Smart Cities: Improving Quality of Life Using ICT & IoT (HONET-ICT), Irbid, Jordan, 2017, pp. 62–67, doi:10.1109/HONET.2017.8102204.
[370] Y. Wang, et al., Application driven network: providing on-demand services for applications, in: SIGCOMM'16, Florianopolis, Brazil, 2016, pp. 617–618, doi:10.1145/2934872.2959075.
[371] M. Monshizadeh, V. Khatri, R. Kantola, Detection as a service: an SDN application, in: Proceedings of the 2017 19th International Conference on Advanced Communication Technology (ICACT), Bongpyeong, South Korea, 2017, pp. 285–290, doi:10.23919/ICACT.2017.7890099.
[372] S. Luo, et al., Context-aware traffic forwarding service for applications in SDN, in: Proceedings of the 2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity), Chengdu, China, 2015, pp. 557–561, doi:10.1109/SmartCity.2015.128.
[373] F. Callegati, W. Cerroni, C. Contoli, R. Cardone, M. Nocentini, A. Manzalini, SDN for dynamic NFV deployment, IEEE Commun. Mag. 54 (10) (2016) 89–95, doi:10.1109/MCOM.2016.7588275.
[374] B. Naudts, W. Tavernier, S. Verbrugge, D. Colle, M. Pickavet, Deploying SDN and NFV at the speed of innovation: toward a new bond between standards development organizations industry fora, and open-source software projects„ IEEE Commun. Mag. 54 (3) (2016) 46–53, doi:10.1109/MCOM.2016.7432171.
[375] N. Akhtar, I. Matta, Y. Wang, Managing NFV using SDN and control theory, in: Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS), Istanbul, Turkey, 2016, pp. 1005–1006, doi:10.1109/NOMS.2016.7502945.
[376] A. Rodriguez-Natal, et al., Global state, local decisions: decentralized NFV for ISPs via enhanced SDN, IEEE Commun. Mag. 55 (4) (2017) 87–93, doi:10.1109/MCOM.2017.1600175.
[377] C.C. Machado, et al., ANSwer: combining NFV and SDN features for network resilience strategies, in: Proceedings of the IEEE Symposium on Computers and Communication (ISCC), Messina, Italy, 2016, pp. 391–396, doi:10.1109/ISCC.2016.7543771.
[378] W. Kellerer, A. Basta, A. Blenk, Using a flexibility measure for network design space analysis of SDN and NFV, in: Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), San Francisco, CA, 2016, pp. 423–428, doi:10.1109/INFCOMW.2016.7562114.
[379] Q. Duan, N. Ansari, M. Toy, Software-defined network virtualization: an architectural framework for integrating SDN and NFV for service provisioning in future networks, IEEE Netw. 30 (5) (2016) 10–16, doi:10.1109/MNET.2016.7579021.
[380] R.-H. Gau, H.-T. Chiu, P.-K. Tsai, Optimizing the service capacity of SDN-based cellular networks with service chaining and NFV, in: Proceedings of the IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Valencia, Spain, 2016, pp. 1–6, doi:10.1109/PIMRC.2016.7794971.
[381] B. Zhang, P. Zhang, Y. Zhao, Y. Wang, X. Luo, Y. Jin, Co-scaler: cooperative scaling of software-defined NFV service function chain, in: Proceedings of the IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Palo Alto, CA, 2016, pp. 33–38, doi:10.1109/NFV-SDN.2016.7919472.
[382] R. Cannistra, Enabling autonomic provisioning in SDN cloud networks with NFV service chaining, in: Optical Fiber Communications Conference and Exhibition (OFC), San Francisco, CA, 2014, pp. 1–3, doi:10.1364/OFC.2014.Tu2I.4.
[383] D. Krishnaswamy, R. Krishnan, D. Lopez, P. Willis, A. Qamar, An open NFV and cloud architectural framework for managing application virality behaviour, in: Proceedings of the 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, NV, 2015, pp. 746–754, doi:10.1109/CCNC.2015.7158071.
[384] A. Khan, A. Zugenmaier, D. Jurca, W. Kellerer, Network virtualization: a hypervisor for the internet? IEEE Commun. Mag. 50 (1) (2012) 136–143, doi:10.1109/MCOM.2012.6122544.
[385] Q. Duan, Y. Yan, A.V. Vasilakos, A survey on service-oriented network virtualization toward convergence of networking and cloud computing, IEEE Trans. Netw. Serv. Manag. 9 (4) (2012) 373–392, doi:10.1109/TNSM.2012.113012.120310.
[386] J. Zhang, W. Xie, F. Yang, An architecture for 5g mobile network based on SDN and NFV, in: Proceedings of the 6th International Conference on Wireless, Mobile and Multi-Media (ICWMMN), Beijing, China, 2016, pp. 87–92, doi:10.1049/cp.2015.0918.
[387] X. Costa-Perez, et al., 5g-crosshaul: an SDN/NFV integrated fronthaul/backhaul transport network architecture, IEEE Wirel. Commun. 24 (1) (2017) 38–45, doi:10.1109/MWC.2017.1600181WC.
[388] M.C. Parker, et al., CHARISMA: converged heterogeneous advanced 5g cloud-RAN architecture for intelligent and secure media access, in: European Conference on Networks and Communications (EuCNC), Athens, Greece, 2016, pp. 240–244, doi:10.1109/EuCNC.2016.7561040.
[389] M. Mechtri, I.G. Benyahia, D. Zeghlache, Agile service manager for 5g, in: Proceedings of the 1st International Workshop on Management of 5G Networks (5GMan), Istanbul, Turkey, 2016, pp. 1285–1290, doi:10.1109/NOMS.2016.7503004.
[390] T. Taleb, A. Ksentini, R. Jäntti, Anything as a service for 5g mobile systems, IEEE Netw. 30 (6) (2016) 84–91, doi:10.1109/MNET.2016.1500244RP.
[391] E. Cau, M. Corici, P. Bellavista, L. Foschini, G. Carella, A. Edmonds, T.M. Bohnert, Efficient exploitation of mobile edge computing for virtualized 5g in EPC architectures, in: Proceedings of the 4th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud), Oxford, 2016, pp. 100–109, doi:10.1109/MobileCloud.2016.24.
[392] A. Ksentini, N. Nikaein, Toward enforcing network slicing on RAN: flexibility and resources abstraction, IEEE Commun. Mag. 55 (6) (2017) 102–108, doi:10.1109/MCOM.2017.1601119.
[393] A. Aissioui, A. Ksentini, A.M. Gueroui, T. Taleb, Toward elastic distributed SDN/NFV controller for 5g mobile cloud management systems, IEEE Access 3 (0) (2015) 2055–2064, doi:10.1109/ACCESS.2015.2489930.
[394] R. Vilalta, et al., Experimental demonstration of distributed multi-tenant cloud/fog and heterogeneous SDN/NFV orchestration for 5g services, in: European Conference on Networks and Communications (EuCNC), Athens, Greece, 2016, pp. 52–56, doi:10.1109/EuCNC.2016.7561003.
[395] F.Z. Yousaf, et al., Network slicing with flexible mobility and qos/qoe support for 5g networks, in: Proceedings of the 4th International Workshop on 5G Architecture, Paris, France, 2017, pp. 1195–1201, doi:10.1109/ICCW.2017.7962821.
[396] T. Ahmed, R. Ferrus, R. Fedrizzi, O. Sallent, Towards SDN/NFV-enabled satellite ground segment systems: bandwidth on demand use case, in: Proceedings of the IEEE International Conference on Communications Workshops (ICC Workshops), Paris, France, 2017, pp. 894–899, doi:10.1109/ICCW.2017.7962772.
[397] R. Ferrus, O. Sallent, T. Ahmed, R. Fedrizzi, Towards SDN/NFV-enabled satellite ground segment systems: end-to-end traffic engineering use case, in: Proceedings of the IEEE International Conference on Communications Workshops (ICC Workshops), Paris, France, 2017, pp. 888–893, doi:10.1109/ICCW.2017.7962771.
[398] S.K. Tayyaba, M.A. Shah, 5g cellular network integration with SDN: Challenges, issues and beyond, international conference on communication, in: Computing and Digital Systems (C-CODE), Islamabad, Pakistan, 2017, pp. 48–53, doi:10.1109/C-CODE.2017.7918900.
[399] C. Bouras, A. Kollia, A. Papazois, SDN & NFV in 5g: advancements and challenges, in: Proceedings of the 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), Paris, France, 2017, pp. 107–111, doi:10.1109/ICIN.2017.7899398.
[400] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J.J. Ramos-Munoz, J. Lorca, J. Folgueira, Network slicing for 5g with SDN/NFV: concepts, architectures, and challenges, IEEE Commun. Mag. 55 (5) (2017) 80–87, doi:10.1109/MCOM.2017.1600935.
[401] L. Gavrilovska1, V. Rakovic1, V. Atanasovski, Visions toward 5g: technical requirements and potential enablers, Wirel. Person. Commun. 87 (3) (2016) 731–757, doi:10.1007/s11277-015-2632-7.
[402] M. Agiwal, A. Roy, N. Saxena, Next generation 5g wireless networks: a comprehensive survey, IEEE Commun. Surv. Tutor. 18 (3) (2016) 1617–1655, doi:10.1109/COMST.2016.2532458.
[403] R. Trivisonno, X. An, Q. Wei, Network slicing for 5g systems: a review from an architecture and standardization perspective, in: Proceedings of the 2017 IEEE Conference on Standards for Communications and Networking (CSCN), Helsinki, Finland, 2017, pp. 36–41, doi:10.1109/CSCN.2017.8088595.
[404] Z. Kotulski, et al., On end-to-end approach for slice isolation in 5g networks, in: Fundamental challenges, 2017 Federated Conference on Computer Science and Information Systems (FedCSIS), Prague, Czech Republic, 2017, pp. 783–792, doi:10.15439/2017F228.
[405] Next generation mobile networks, 2015, [Online]. Available: https://www.ngmn.org/fileadmin/ngmn/content/downloads/Technical/2015/NGMN_5G_White_Paper_V1_0.pdf.
[406] X. Li, et al., Network slicing for 5g: challenges and opportunities, IEEE Internet Comput. 21 (5) (2017) 20–27, doi:10.1109/MIC.2017.3481355.
[407] R. Pries, et al., Network as a service - a demo on 5g network slicing, in: Proceedings of the 2016 28th International Teletraffic Congress (ITC 28), 1, Germany, 2016, pp. 209–211, doi:10.1109/ITC-28.2016.136.
[408] R. Ravindran, et al., 5g-ICN: delivering ICN services over 5g using network slicing, IEEE Commun. Mag. 55 (5) (2017) 101–107, doi:10.1109/MCOM.2017.1600938.
[409] J.J.P. Manrese, et al., Dynamic qos/qoe assurance in realistic NFV-enabled 5g access networks, in: Proceedings of the 2017 19th International Conference on Transparent Optical Networks (ICTON), Girona, Spain, 2017, pp. 1–4, doi:10.1109/ICTON.2017.8025149.
[410] J.M. Tilli, R. Kantola, Data plane protocols and fragmentation for 5g, in: Proceedings of the 2017 IEEE Conference on Standards for Communications and Networking (CSCN), Helsinki, Finland, 2017, pp. 207–213, doi:10.1109/CSCN.2017.8088623.
[411] S. Li, et al., Protocol oblivious forwarding (POF): software-defined networking with enhanced programmability, IEEE Netw. 31 (2) (2017) 58–66, doi:10.1109/MNET.2017.1600030NM.
[412] D. Kutscher, It's the network: towards better security and transport performance in 5g, in: Proceedings of the 2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), San Francisco, CA, USA, 2016, pp. 656–661, doi:10.1109/INFCOMW.2016.7562158.
[413] J.V. Reyes, et al., IP traffic classification in NFV: a benchmarking of supervised machine learning algorithms, in: Proceedings of the 2017 IEEE Colombian Conference on Communications and Computing (COLCOM), Cartagena, Colombia, 2017, pp. 1–6, doi:10.1109/ColComCon.2017.8088199.
[414] Huawei white paper, 5g a technology vision, 2014. [Online]. Available: http://www.huawei.com/5gwhitepaper/.
[415] J. Lv, X. Wang, K. Ren, M. Huang, K. Li, ACO-inspired information-centric networking routing mechanism, Comput. Netw. 126 (2017) 200–217, doi:10.1016/j.comnet.2017.07.004.
[416] S. Li, Y. Zhang, D. Raychaudhuri, R. Ravindran, A comparative study of mobil-

ityfirst and NDN based ICN-iot architectures, heterogeneous networking for quality, reliability, security and robustness (QShine), in: 10th International Conference on, Rhodes, 2014, pp. 158–163, doi:10.1109/QSHINE.2014.6928680.
[417] Y. Nishiyama, M. Ishino, Y. Koizumi, T. Hasegawa, K. Sugiyama, A. Tagami, Proposal on routing-based mobility architecture for ICN-based cellular networks, in: Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), San Francisco, CA, USA, 2016, pp. 467–472, doi:10.1109/INFCOMW.2016.7562122.
[418] R. Ravindran, X. Liu, A. Chakraborti, X. Zhang, G. Wang, Towards software defined ICN based edge-cloud services, in: Proceedings of the IEEE 2nd International Conference on Cloud Networking (CloudNet), San Francisco, CA, 2013, pp. 227–235, doi:10.1109/CloudNet.2013.6710583.
[419] J. Ren, et al., On the deployment of information-centric network: programmability and virtualization, in: Proceedings of the International Conference on Computing, Networking and Communications (ICNC), Garden Grove, CA, 2015, pp. 690–694, doi:10.1109/ICCNC.2015.7069429.
[420] A.F.R. Trajano, M.P. Fernandez, ContentSDN: a content-based transparent proxy architecture in software-defined networking, in: Proceedings of the IEEE 30th International Conference on Advanced Information Networking and Applications (AINA), Crans-Montana, 2016, pp. 532–539, doi:10.1109/AINA.2016.103.
[421] S. Salsano, N. Blefari-Melazzi, F.L. Presti, G. Siracusano, P.L. Ventre, Generalized virtual networking: an enabler for service centric networking and network function virtualization, in: Proceedings of the 16th International Telecommunications Network Strategy and Planning Symposium, Funchal, Portuga, 2014, pp. 1–7, doi:10.1109/NETWKS.2014.6958523.
[422] K. Ueda, K. Yokota, J. Kurihara, A. Tagami, Towards the NFVI-assisted ICN: integrating ICN forwarding into the virtualization infrastructure, in: IEEE Global Communications Conference (GLOBECOM), Washington, DC, 2016, pp. 1–6, doi:10.1109/GLOCOM.2016.7842348.
[423] M. Arumaithurai, J. Chen, E. Monticelli, X. Fu, K.K. Ramakrishnan, Exploiting ICN for flexible management of software-defined networks, in: Proceedings of the 1st International Conference on Information-centric Networking, Pairs, France, 2014, pp. 107–116, doi:10.1145/2660129.2660147.
[424] M. Arumaithurai, J. Chen, E. Maiti, X. Fu, K.K. Ramakrishnan, Prototype of an ICN based approach for flexible service chaining in SDN, in: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Hong Kong, 2015, pp. 5–6, doi:10.1109/INFCOMW.2015.7179315.
[425] J. Ren, et al., On the deployment of information-centric network: Programmability and virtualization, in: Proceedings of the 2015 International Conference on Computing, Networking and Communications (ICNC), Garden Grove, CA, USA, 2015, pp. 690–694, doi:10.1109/ICCNC.2015.7069429.
[426] R. Ravindran, et al., 5g-ICN : delivering ICN services over 5g using network slicing, IEEE Commun. Mag. 55 (5) (2017) 101–107, doi:10.1109/MCOM.2017.1600938.
[427] S. Chen, H. Xu, D. Liu, B. Hu, H. Wang, A vision of iot: applications, challenges, and opportunities with china perspective, IEEE Internet Things J. 1 (4) (2014) 349–359, doi:10.1109/JIOT.2014.2337336.
[428] K. Rajaram, G. Susanth, Emulation of iot gateway for connecting sensor nodes in heterogenous networks, in: Proceedings of the 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP), Chennai, India, 2017, pp. 1–5. doi: 10.1109/ICCCSP.2017.7944105.
[429] F.Y. Wang, et al., Network softwarization and parallel networks: beyond software-defined networks, IEEE Netw. 30 (4) (2016) 60–65, doi:10.1109/MNET.2016.7513865.
[430] B. Dhanalaxmi, G.A. Naidu, A survey on design and analysis of robust iot architecture, in: Proceedings of the 2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bangalore, India, 2017, pp. 375–378, doi:10.1109/ICIMIA.2017.7975639.
[431] E. Yigitoglu, L. Liu, M. Looper, C. Pu, Distributed orchestration in large-scale iot systems, in: Proceedings of the 2017 IEEE International Congress on Internet of Things (ICIOT), Honolulu, HI, USA, 2017, pp. 58–65, doi:10.1109/IEEE.ICIOT.2017.16.
[432] S. Chen, H. Xu, D. Liu, B. Hu, H. Wang, A vision of iot: applications, challenges, and opportunities with china perspective, IEEE Internet Things J. 1 (4) (2014) 349–359, doi:10.1109/JIOT.2014.2337336.
[433] S. Baidya, M. Levorato, Edge-assisted content and computation-driven dynamic network selection for real-time services in the urban iot, in: Proceedings of the 2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Atlanta, GA, USA, 2017, pp. 796–801, doi:10.1109/INFCOMW.2017.8116478.
[434] F. Marino, et al., Towards softwarization in the iot: integration and evaluation of t-res in the onem2m architecture, in: Proceedings of the 2017 IEEE Conference on Network Softwarization (NetSoft), Bologna, Italy, 2017, pp. 1–5, doi:10.1109/NETSOFT.2017.8004202.
[435] S. Fichera, M. Gharbaoui, P. Castoldi, B. Martini, A. Manzalini, On experimenting 5g: testbed set-up for SDN orchestration across network cloud and iot domains, in: Proceedings of the 2017 IEEE Conference on Network Softwarization (NetSoft), Bologna, Italy, 2017, pp. 1–6, doi:10.1109/NETSOFT.2017.8004245.
[436] H. Khazaei, H. Bannazadeh, A.L. Garcia, End-to-end management of iot applications, in: Proceedings of the 2017 IEEE Conference on Network Softwarization (NetSoft), Bologna, Italy, 2017, pp. 1–3, doi:10.1109/NETSOFT.2017.8004252.
[437] B. Xiao, et al., Intelligent data-intensive iot: A survey, in: Proceedings of the 2016 2nd IEEE International Conference on Computer and Communications (ICCC), Chengdu, China, 2016, pp. 2362–2368, doi:10.1109/CompComm.2016.7925122.
[438] IRTF, information-centric networking (ICN) research challenges, 2016, [Online]. Available: https://tools.ietf.org/html/rfc7927.
[439] HP business white paper, iot to drive NFV adoption, 2016, Pp. 1-8, [Online]. Available: https://h20195.www2.hpe.com/V2/getpdf.aspx/4AA6-6474ENW.pdf.
[440] R. Vilalta, et al., End-to-end SDN orchestration of iot services using an SDN/NFV-enabled edge node, in: Optical Fiber Communications Conference and Exhibition (OFC), Anaheim, CA, 2016, pp. 1–3, doi:10.1364/OFC.2016.W2A.42.
[441] P. Du, P. Putra, S. Yamamoto, A. Nakao, A context-aware iot architecture through software-defined data plane, in: IEEE Region 10 Symposium (TENSYMP), Bali, Indonesia, 2016, pp. 315–320, doi:10.1109/TENCONSpring.2016.7519425.
[442] F.V. Abeele, J. Hoebeke, G.K. Teklemariam, I. Moerman, P. Demeester, Sensor function virtualization to support distributed intelligence in the internet of things, Wirel. Person. Commun. 81 (4) (2015) 1415–1436, doi:10.1007/s11277-015-2481-4.
[443] A.B.G. Hernando, et al., Virtualization of residential iot functionality by using NFV and SDN, in: Proceedings of the IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, 2017, pp. 86–87. doi: 10.1109/ICCE.2017.7889240.
[444] M.A. Salahuddin, A. Al-Fuqaha, M. Guizani, K. Shuaib, F. Sallabi, Softwarization of internet of things infrastructure for secure and smart healthcare, Computer 50 (7) (2017) 74–79, doi:10.1109/MC.2017.195.
[445] A. Hegyi, H. Flinck, I. Ketyko, P. Kuure, C. Nemes, L. Pinter, Application orchestration in mobile edge cloud, in: Proceedings of the IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W), Augsburg, Germany, 2016, pp. 230–235, doi:10.1109/FAS-W.2016.56.
[446] N. Bizanis, F.A. Kuipers, SDN and virtualization solutions for the internet of things: a survey, IEEE Access 4 (0) (2016) 5591–5606, doi:10.1109/ACCESS.2016.2607786.

Bo Yi received the B.S. degree and the M.S. degree in computer science from the South-Central University for Nationalities, Wuhan, China in 2012 and 2015 respectively. He is currently working toward the Ph.D degree in the Northeastern University, Shenyang, China. His research interests include routing and service function chain in SDN and NFV, etc.

Xingwei Wang received the B.S., M.S., and Ph.D. degrees in computer science from the Northeastern University, Shenyang, China in 1989, 1992, and 1998 respectively. He is currently a Professor at the College of Computer Science and Engineering, Northeastern University, Shenyang, China. His research interests include cloud computing and future Internet, etc. He has published more than 100 journal articles, books and book chapters, and refereed conference papers. He has received several best paper awards.

Keqin Li is a SUNY Distinguished Professor of computer science. His research interests include parallel computing and high-performance computing, distributed computing, energy-efficient computing and communication, heterogeneous computing systems, cloud computing, big data computing, CPU-GPU hybrid and cooperative computing, multicore computing, storage and file systems, wireless communication networks, sensor networks, peer-to-peer file sharing systems, mobile computing, service computing, Internet of things and cyber-physical systems. He has published over 470 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He is currently or has served on the editorial boards of IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers, IEEE Transactions on Cloud Computing, IEEE Transactions on Services Computing, IEEE Transactions on Sustainable Computing. He is an IEEE Fellow.

Sajal K. Das is the Chair of Computer Science Department and Daniel St. Clair Endowed Chair at the Missouri University of Science and Technology, Rolla. During 2008–2011, he served the US National Science Foundation as a Program Director in the Division of Computer Networks and Systems. His current research interests include wireless and sensor networks, mobile and pervasive computing, big data, cyber-physical systems, smart healthcare, distributed and cloud computing, security and privacy, biological and social networks, applied graph theory and game theory. He published more than 600 research articles in high quality journals and refereed conference proceedings. He holds 5 US patents, coauthored 51 book chapters and four books titled Smart Environments: Technology, Protocols, and Applications (2005), Handbook on Securing Cyber-Physical Critical Infrastructure: Foundations and Challenges (2012), Mobile Agents in Distributed Computing and Networking (2012), and Principles of Cyber-Physical Systems (2016). His h-index is 70 with more than 20,000 citations according to Google Scholar. He received 10 Best Paper Awards in prestigious conferences such as ACM MobiCom'99, IEEE PerCom'06 and IEEE SmrtGridComm'12. He serves as the founding Editor-in-Chief of the Pervasive and Mobile Computing journal, and as Associate Editor of IEEE Transactions on Mobile Computing, ACM Transactions on Sensor Networks, and several others. He is a Fellow of the IEEE.

Min Huang received the B.S. degree in automatic instrument, the M.S. degree in systems engineering, and Ph.D. degree in control theory from the Northeastern University, Shenyang, China in 1990, 1993, and 1999 respectively. She is currently a Professor at the College of Information Science and Engineering, Northeastern University, Shenyang, China. Her research interests include modeling and optimization for logistics and supply chain system, etc. She has published more than 100 journal articles, books, and refereed conference papers.